

Г. А. Кардашев

Цифровая электроника на персональном компьютере

Москва
Горячая линия - Телеком
2003

ББК 32 844
К21

Кардашев Г.А.

К21 Цифровая электроника на персональном компьютере Electronics Work bench и Micro-Cap – М Горячая линия–Телеком, 2003 – 311 с ил – (Массовая радиобиблиотека, 1263)

ISBN 5-93517-140-6

Дается введение в схемотехническое моделирование цифровых электронных устройств на компьютере Моделирование выполняется с использованием наиболее простых и популярных программ Electronics Workbench и Micro-Cap Подробно излагается методика компьютерного моделирования цифровых устройств от простейших логических элементов до микропроцессора Последовательно с рассмотрением работы моделей приводятся необходимые сведения о программах и советы по их конкретному применению Книга может быть использована для изучения и практического применения цифровой электроники и методов схемотехнического моделирования электронных устройств на компьютерах.

Для широкого круга читателей

ББК 32.844

Адрес издательства в Интернет www techbook.ru
e-mail radios_hl@mtu-net.ru

ISBN 5-93517-140-6

Предисловие

Я старался насколько мог и умел отделаться от трудности и скуки вычислений, докучность которых обычно отпугивает весьма многих
Дж. Непер (шотл математик XVI в – изобретатель логарифмов)

Создание компьютера следует считать наивысшим достижением человечества за прошедшее время, поскольку нет ни одной сферы деятельности от любого производства до разнообразного творчества, где бы с его помощью не происходила настоящая революция В первую очередь с нарастающей интенсивностью продолжается «генерирование» самих компьютеров и, с появлением их новых поколений, словно грибы после хорошего теплого дождика происходят открытия в генетике и физике, создаются интеллектуальные производства и «умные» дома В основе этой информационно-интеллектуальной революции лежит прогресс в цифровой микроэлектронике

В данной книге популярно изложены методы компьютерного моделирования основных цифровых устройств Книга является, продолжением предыдущей, посвященной моделированию аналоговых устройств (см Кардашев Г. А «Виртуальная электроника») Изложение ведется от «простого – к сложному», и в последующих разделах активно используются предыдущие Последовательно с изложением нового материала поясняется смысл основных научно-технических терминов и понятий цифровой электроники Однако настоящее издание не является ни систематическим учебником по цифровой электронике, ни справочником по компьютерным программам, а лишь практическим введением в данную область.

Рассматриваемая в книге компьютерная схемотехника, как основа виртуальной электроники, в программном обеспечении является разновидностью САПР (Систем Автоматизированного ПРоектирования) электронных устройств. В настоящее время разработано много подобных программ «Естественный отбор» среди них привел к выбору двух наиболее популярных и «легких», но достаточно эффективных Electronics Workbench («Электронная лаборатория») и Micro-Cap (Microcomputer Circuit Analysis Program – «Программа анализа схем на микрокомпьютерах»)

Далее мы будем называть их сокращенно программа **EWB** и **MC** соответственно, выделяя полужирным шрифтом для более быстрой навигации в тексте Основу книги составляют экранные снимки отлаженных схем, окон настроек виртуальных приборов для их исследования и результатов работы цифровых устройств В связи с особенностями используемых программ наблюдается некоторое, не влияющее на содержание, различие в графике приведенных схем Приводимые примеры, как правило, задублированы в обеих программах

При проведении моделирования не применяется какая-либо «сложная математика», а широко используются сервисные возможности самих компьютерных программ Для осмысленного применения программ по мере необходимости даются азы булевой алгебры и некоторые смежные вопросы из курса информатики

Предполагается, что читатели обладают некоторыми навыками работы на ПК в Windows, и будут работать с книгой не «всухую», а параллельно с набором схем на компьютере Тогда они обнаружат в программах кнопки Color (цвет) и смогут на свой вкус раскрасить проводники, лучи осциллограмм, фон и шрифты, используя богатую цветовую палитру виртуального мира

Техника использования программ приводится непосредственно при ее применении к конкретным схемам Изложение сопровождается советами, позволяющими успешно использовать программы и избежать возможных ошибок На созданных компьютерных моделях можно, изменять в широких пределах как состав и параметры компонентов, так и собственно схемотехнику, отбирая для практического изготовления наилучшие варианты Это, безусловно, будет продуктивно во всех отношениях от минимизации временных и финансовых затрат, до развития знаний, умений и навыков, составляющих основу профессионализма

Цель книги заключается не только в том, чтобы познакомить читателей с основами цифровой электроники и новым эффективным методом познания – компьютерным моделированием, но и в том, чтобы пробудить у них горячее желание побыстрее установить на компьютере моделирующие программы и погрузиться в прекрасный мир цифровой электроники с помощью ее основного детища – компьютера

1. ОСНОВЫ ЦИФРОВОЙ ЭЛЕКТРОНИКИ

1.1. Моделирование логики простейших цепей

Быть или не быть вот в чем вопрос
В Шекспир

Безусловным фаворитом цифровой электроники является компьютер Как и человечество, компьютеры имеют свою историю и родословную механические, электромеханические, электронные, оптические, биологические

Компьютер обладает сверхфеноменальной памятью, неотразимой эстетической красотой пользовательского интерфейса и богатейшими возможностями Однако, несмотря на всю сложность современных компьютеров, они в своей основе прозаично «перемалывают» эти гамлетовские «быть или не быть» в виде электрических сигналов – «единиц и нулей», правда, в громадном количестве, с бешеной скоростью и в необходимом порядке

В электрических контактных схемах можно ясно увидеть генетические основы, из которых произошли ЭВМ

У итальянского писателя Джанни Родари среди сказок с несколькими концами есть сказка о волшебнике Вклю-Чу Помните, как он пришел вечером в гости к одной синьоре и хотел продемонстрировать ей чудеса с помощью своей волшебной палочки Не тут-то было! Синьора, обыденными действиями, включив свет, телевизор и поговорив по телефону, буквально обескуражила бедного волшебника Нетрудно вообразить, что все эти «чудеса» померкли бы, если бы описанные события происходили в наше время, а синьора продемонстрировала волшебнику персональный компьютер, укомплектованный мультимедиа и выходом в Интернет Сказка имеет несколько окончаний

В «Первом конце» к этой сказке «Волшебник решил понять, как устроен этот мир » Похвальное желание Поставим себя на его место, и для того чтобы разобраться в цифровом мире, обратимся к простым примерам, выполняя их с помощью компьютера Начнем с элементарного в виртуальном компьютерном пространстве смоделируем это заветное действие «Вклю-Чу»

Разработка модели в программе Electronics Workbench

Электрическая цепь обыкновенной лампы состоит из источника питания, соединительных проводов, выключателя и самой лампы Поскольку в модели нас интересуют только факты включения и выключения, то абстрагируемся (отвлечемся) от всех второстепенных, с точки зрения поставленной цели моделирования, характеристик реального объекта

Выберем в качестве компонентов модельной цепи источник питания – идеальный источник постоянного напряжения с неизменной ЭДС и внутренним сопротивлением равным нулю, идеальные соединительные провода, также с сопротивлением равным нулю, лампу накаливания с неизменным сопротивлением и, наконец, идеальный выключатель (идеальный ключ) Под последним будем подразумевать устройство с бесконечно большим сопротивлением в выключенном и нулевым во включенном состоянии, имеющее бесконечно малое время переключения из одного состояния в другое Отобразим модельную электрическую цепь в виде схемы на компьютере, воспользовавшись программой **EWB**

В этой программе реализован стандартный многооконный интерфейс с ниспадающими и разворачивающимися меню При установке (инсталляции) программы (Setup) в окне выбора компонентов (Select Components) желательно выбрать европейский стандарт DIN (Deutsche Ingenieuring Normen – немецкий инженерный стандарт), к которому ближе российские ГОСТы (рис 1)

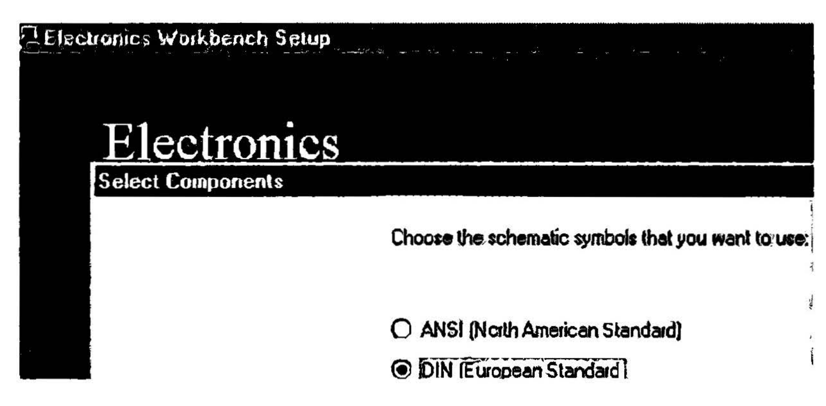


Рис 1 Окно установки стандарта УГО (**EWB**)

В противном случае ряд компонентов окажется с неузнаваемыми УГО (условно графическими обозначениями), например, резистор будет похож на острозубую пилу Если программа уже установлена, то переход от одних стандартов к другим можно выполнить в окне «Свойства». «Ярлык», далее «Объект» и в командной строке допечатать после EXE через пробел \DIN Можно также создать два ярлыка и для одного сделать адрес \DIN, а для другого \ANSI Последний представляет собой американский стандарт (ANSI – American National Standard Institute), чаще используемый в англоязычной и переводной литературе

На рис 2 показан общий вид рабочего окна **EWB** с рядом открытых в левой половине экрана панелей для выбора библиотечных компонентов и некоторыми компонентами из них в правой половине экрана

Откроем на панели компонентов пиктограмму группы Source (источники) и выберем в нем Battery (батарея) Удерживая ЛКМ (левую кнопку мыши) в нажатом состоянии, перетаскиваем пиктограмму с УГО батареи на свободную часть рабочей области экрана и отпускаем ЛКМ (рис 2) Эту процедуру, аналогичную перемещению при редактировании в Word выделенных фрагментов текста или изображений, принято называть буксировкой.

Аналогично, переносим на экран из раздела Basic (основные компоненты) Switch (переключатель) и Ground (заземление), а затем, из раздела Indicators (индикаторы) компонент Bulb (лампа накаливания) На этом первая часть «строительства» схемы закончена: «рабочие материалы доставлены на стройплощадку»

Теперь надо соотнести эти компоненты с необходимой схемой их соединения. Батарея и лампа являются электрическими двухполюсниками, а вот переключатель имеет три вывода и другие особенности В целях удобства объяснения его работы и дальнейшего «монтажа» разметим его выводы. Для этого вновь обратимся к панели Basic и извлечем из нее на рабочее поле Connector (соединитель), представляющий на схемах УГО

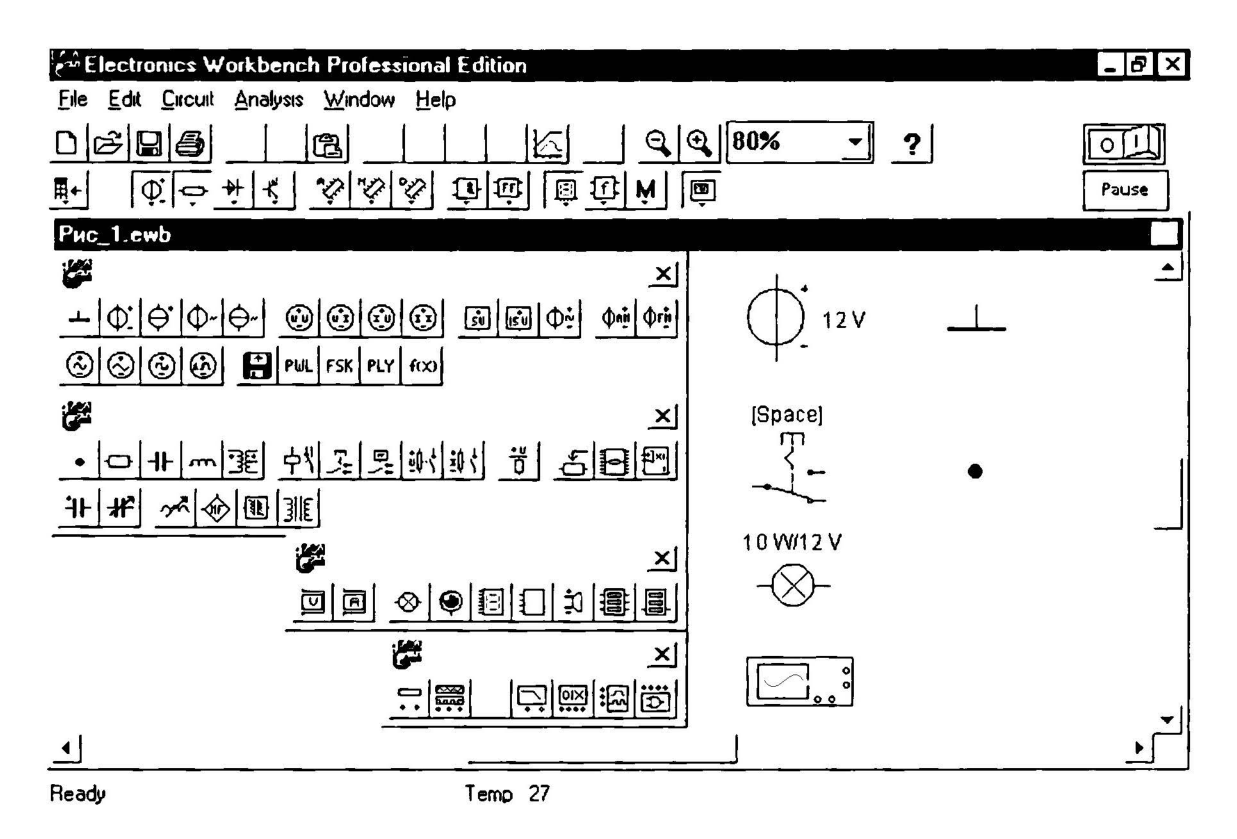


Рис 2 Основное окно программы EWB с дополнительными окнами выбора компонентов

неразъемное соединение (узел) Эту процедуру повторим еще два раза, размещая узлы на некотором расстоянии от выводов переключателя (рис 3)

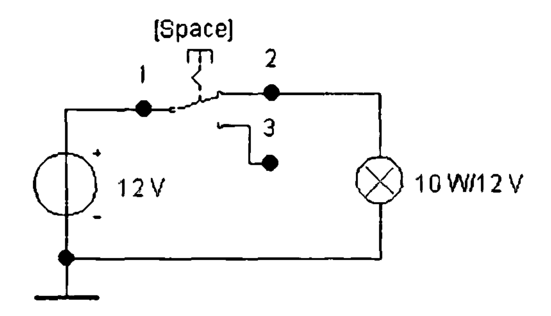


Рис 3 Схема-модель лампы с выключателем (EWB)

В каждом узле могут соединяться не более четырех ортогональных проводников На местах этих возможных соединений при попадании на них острия стрелки курсора возникает дополнительное утолщение, напоминающее капельку олова при обычной пайке радиодеталей Этот этап нашего «общения» с виртуальным пространством как бы сопровождается сигналами «Соединилось? – Нет, не соединилось!» или, наконец, «Соединилось? – Да, соединилось!» Из места возникшего соединения, нажав ЛКМ и перемещая курсор по экрану в нужном направлении, можно вытянуть проводник и довести его до необходимого вывода другого компонента

Здесь также возникнет монтажный узел После этого ЛКМ надо отпустить На экране должно возникнуть изображение проводника Если изображение проводника не осталось на экране, после того как курсор был свободно перемещен в произвольное место экрана, процедуру придется повторить Соединение можно выполнить из любого вывода каждого схемного компонента, не забывая при этом, что всего в одном узле возможно осуществление только четырех определенных соединений, причем одно из них уже может быть занято самим компонентом

Графический редактор EWB позволяет внести в схему необходимые изменения, если это потребуется Например, когда выбранное место «пайки» оказалось неудачным, проводник можно удалить после его однократного выделения ЛКМ Можно также, выделив область соединения курсором, нажать ЛКМ и, удерживая

конец проводника, перенести его до соединения в новом месте Если на выделенном проводнике нажать ЛКМ, то после возникновения двойной стрелки эту часть проводника можно отбуксировать в указанных направлениях для придания схеме необходимого вида

Совет **(EWB)** Для успешного позиционирования мест соединений удобнее работать при больших масштабах изображения, например 100% Начертив часть схемы, можно возвращаться в более удобный режим скажем 80% Не старайтесь располагать соединительные узлы близко к компонентам После того как соединение выполнено, узел, компонент или проводник можно выделить и переместить стрелками на клавиатуре в нужное место

Двойной щелчок по узлу вызывает окно Connector Properties (свойства проводника) и позволяет снабдить его Label – текстовой меткой из 20 символов (к сожалению, англоязычных)

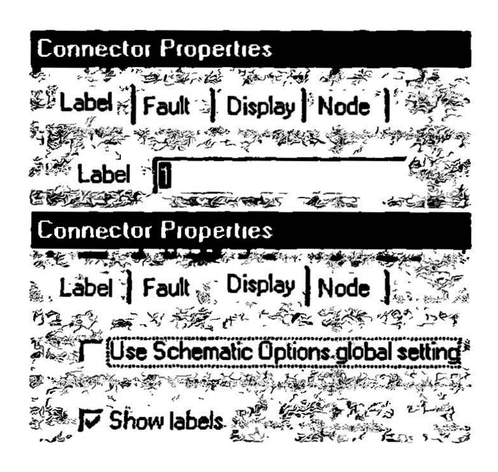


Рис. 4. Окна установки метки узла и имени управляющей клавиши (EWB)

На рис 4 показана информативная часть окна при присвоении метки и выбор ее отображения на дисплее (Show labels). Например, на рис 2 выводам переключателя присвоены цифровые метки 1, 2, 3 Если какая-либо метка на экране накладывается на другие элементы схемы, то следует прибегнуть к описанным вы-

ше приемам графического редактирования. Данные метки не следует путать со сквозной расчетной нумерацией узлов (Node), начинающейся с 0 – заземление. Эта нумерация при моделировании выполняется программой автоматически и ее можно выявить, активируя соответствующий узел и нажав в появившемся окне Connector Properties на кнопку Node Машинные номера используются внутри программы при моделировании поведения цепей в опции Analysis.

Теперь подробнее рассмотрим работу переключателя (см. рис. 3), представляющего собой механически управляемую кнопку без автоматического возврата в исходное положение (в отличие, например, от обычной звонковой кнопки), с двумя одновременно срабатывающими ключами (контактами), имеющими один общий вывод. В отсутствие внешнего воздействия один ключ замкнут, а другой – разомкнут В виртуальной электрической цепи этот переключатель дает возможность путем нажатия на клавиатуре по клавише, имя которой заключено в УГО в квадратные скобки, перевести соединение от 1–2 к 1–3, или наоборот. При этом на экране возникает картина анимации работы переключателя. По умолчанию имя управляющей клавиши [Space], чтобы его заменить, надо выполнить двойной щелчок ЛКМ по УГО переключателя или однократный ПКМ и затем выбрать Component Properties. Далее в появившемся окне Switch Properties (свойства переключателя) в позиции Value (значение) впечатать (при английской раскладке клавиатуры) напротив Key (клавиша) выбранный символ, например X (см. рис. 4). Квадратные скобки печатать не следует: они возникнут на экране автоматически (см. рис. 3). Переключатель будет управляться выбранной клавишей независимо от регистра буквы заглавная или прописная, т.е. дополнительная клавиша Ctrl не действует.

Упорядочим расположение выбранных компонентов на экране, если оно не соответствует воображаемой схеме Для этого ЛКМ выделяем необходимый компонент (при этом он примет активный красный вид, а курсор – вид руки) и перемещаем (буксируем) его в нужное положение Возможно, на этом этапе, потребуется изменить пространственную ориентацию компонентов В данном конкретном случае удобнее повернуть лампу на 90° против часовой стрелки выделим лампу (однократным нажатием ЛКМ) и нажмем на кнопку (пиктограмму) Rotate (вращение) в горизонтальном ряду инструментов Эту же операцию можно провести с клавиатуры, выделив лампу и нажав Control+R (разу-

меется, находясь при английской раскладке клавиатуры) Можно также после выделения компонента, войдя в меню Circuit (схема), воспользоваться командой Rotate

Далее выполняем соединения компонентов Лучше всего, как и при сборке реальных цепей, начать с положительного полюса «+» батареи Устанавливаем стрелку курсора в верхнюю часть вывода там появляется жирная черная точка – символ неразъемного соединения Нажимаем ЛКМ и кратчайшим путем ведем линию-резинку к крайнему выводу 1 переключателя После того как там возникнет символ соединения, отпускаем ЛКМ (рис 3)

На экране возникает изображение соединительного проводника в виде двух ортогональных отрезков Поскольку в цепи управления лампой нам достаточно иметь режим замыкания или размыкания одной пары выводов (одного ключа), то выберем, например, выводы, обозначенные как 1 – 2 Аналогично соединяем вывод 2 переключателя с верхним выводом лампы и ее нижний вывод с отрицательным полюсом «–» батареи Вывод 3 переключателя оставляем незадействованным (при переключении в эту позицию лампа выключается)

При монтаже схем часто возникают различные графические ошибки настоящие и кажущиеся из-за несовершенства программного графического редактора, особенно при изменении масштаба, выводимого на экран изображения Наиболее сложно отыскать подобные ошибки при соединении двух компонентов, так как при трех и четырех обязательно должен возникнуть монтажный узел, а его отсутствие сразу заметно Для обнаружения подобных артефактов (от лат arte – искусственно + factus – сделанный) необходимо провести пробные буксировки одного из соединяемых компонентов он должен «тянуть» за собой соединительные провода Можно также перейти к очень большим увеличениям, но для сложных схем это не очень удобно

Совет (EWB) Не пытайтесь соединять выводы компонентов непосредственно друг с другом без использования соединительных проводников и монтажных узлов возникнет ложная видимость соединения, приводящая к ошибкам моделирования или неработоспособным схемам

После того как общий чертеж принципиальной схемы выполнен (см рис 3), надо отредактировать параметры (свойства) компонентов

Начинаем с батареи Дважды щелкаем на ней ЛКМ На экране (рис 5) появляется подменю Battery Properties (свойства батареи)

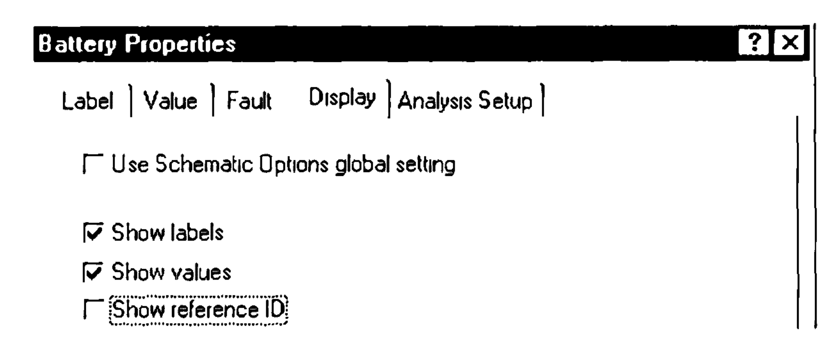


Рис 5 Окно установки свойств батареи (EWB)

Выбираем в этом подменю Label (обозначения) и печатаем буквенный символ ЭДС E, затем Value (значение) и набираем в соответствующем окошке цифру 5, оставляя единицу измерения V, т е Вольт Далее в позиции Display (дисплей) отмечаем галочкой Show labels и Show values (показать на дисплее метку и величину), подтверждаем сделанный выбор свойств нажатием на кнопку **OK**

Переходим к лампе Действуя аналогично предыдущему, выделяем лампу, вызывая диалоговое окно для редактирования ее параметров (рис 6)

Набираем в окошке Label «Lamp» Устанавливаем в позиции Value PMAX (максимальная мощность) 10 W (Ватт) Здесь же набираем в окошке VMAX (максимальное напряжение) 5

Совет Следите за разделителем целой и дробной части десятичного числа в обеих программах – это точка Обращайте внимание на единицы измерения и при необходимости переходите к кратным, например kW (кВт), mV (мВ) и т п

Выбор численного значения параметров можно сделать для другой конкретной или воображаемой батареи и лампочки Отредактированная схема-модель лампы показана на рис 7

Bulb Properties ? X

Label | Value | Fault | Display

Label: Lamp

Reference ID:

Bulb Properties ? X

Label | Value | Fault | Display

Maximum power (PMAX) 10 W

Maximum voltage (VMAX) 5 V

Bulb Properties ? X

Label | Value | Fault | Display

Use Schematic Options global setting

Show labels

Show values

Show reference ID

Рис 6 Окно установки свойств лампы (**EWB**)

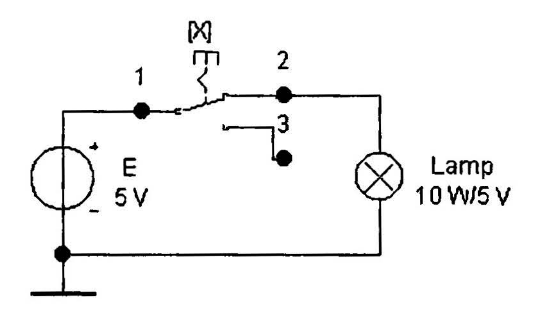


Рис 7 Отредактированная схема-модель лампы (**EWB**)

Проверка работоспособности собранной схемы

Устанавливаем в виртуальном выключателе Activate simulation (включение моделирования) [O I], размещенном в правой верхней части панели инструментов, указатель на I (In – включено), и делаем щелчок ЛКМ Клавиша этого выключателя переходит в положение «включено» Прерывание моделирования производится нажатием на расположенную ниже кнопку [Pause] Pause (пауза), повторное нажатие отменяет эту команду Выключение моделирования производится нажатием на O (Out – выключено) Эти же процедуры можно осуществить и из меню Analysis Activate, Pause, Stop или с клавиатуры Control+G, F9, Control+T

После запуска моделирования переводим выключатель [X] на схеме (рис 7) в положение «включено» (нажав на клавишу буквы X при английской раскладке клавиатуры) и наблюдаем, как лампочка окрашивается в черный цвет (имитация ее горения) Нажимая несколько раз на [X], как бы включаем и выключаем лампу

Рассматриваемая цепь, конечно, является аналоговой, но наличие в ней ключевого элемента, с которого мы начали наш рассказ, позволит позже перейти к цифровым устройствам

Логические состояния

Различают аналоговую и дискретную формы описания поведения систем и представления информации Аналоговые сигналы имеют непрерывную зависимость напряжения (или тока) во времени, аналогичную соответствующим физическим макропроцессам, цифровые – являются дискретными по времени, а также квантованными по уровню

Основным «кирпичиком» цифровых сигналов и «порцией» информации о состоянии систем, в которых они наблюдаются, является **бит** (по-английски **bit** – **bi**nary dig**it** – двоичная цифра) В компьютерах вся информация передается, обрабатывается и хранится побитно Вот, поистине, «бит или не бит»

Так уж получилось, что именно на родине Шекспира два века назад родился гениальный математик-самоучка Джордж Буль Школьные учебники по математике привели юношу буквально в ужас своей нестрогостью и нелогичностью, и он обратился к трудам классиков Лапласа и Лагранжа Переосмысливая основы математики, он создал алгебру логики, называемую теперь

булевой алгеброй Современные школьники знакомятся с ее азами в курсе «Информатика» Возможно, что среди них найдется и тот, кто обнаружит в ее структуре особенности, которые позволят ему также сказать новое слово в науке

Применим булеву алгебру к контактным цепям, состоящим из идеальных двухполюсников Логическое состояние любого такого двухполюсника можно представить одной переменной, которая может принять только два возможных значения «истинно» или «ложно»

Выберем в качестве параметра электрической цепи, с помощью которого можно определить эти состояния, уровень напряжения Он также может принять только два значения *высокий уровень* и *низкий уровень* Сокращенно на русском языке *в* и *н*, на английском языке, соответственно, High (H) и Low (L)

Пусть в электрической цепи *замкнутое состояние ключа – «истинно»*, тогда *разомкнутое состояние ключа – «ложно»*, сокращенно на русском языке *и* и *л* На английском языке, соответственно, Truth (T) и False (F) При принятых соглашениях в схеме на рис 7 состоянию «истинно» соответствует замкнутый ключ 1–2 и высокий уровень напряжения на лампе (она горит), а разомкнутому ключу 1–3 состояние – «ложно» низкий уровень напряжения на лампе (она не горит)

Покажем эти состояния на осциллоскопе (осциллографе), имеющим в программе **EWB** специальную пиктограмму Oscilloscope в группе инструментов (Instruments) Отбуксировав его пиктограмму на рабочее поле, получаем свернутое изображение виртуального *схемного* прибора, служащего для включения в схему (рис 8)

Здесь активными являются четыре клеммы две нижние – входы каналов A и B, и две верхние – заземление и внешняя синхронизация развертки Для вызова картины полной лицевой панели осциллоскопа необходимо дважды щелкнуть ЛКМ по его свернутому изображению На лицевой панели виртуального осциллоскопа (рис 8) с помощью ЛКМ можно провести необходимые предустановки режимов работы прибора, а также наблюдать на экране временные зависимости процессов в собранных схемах. Более подробное описание осциллоскопа (правда, на английском языке) можно получить, щелкнув ПКМ по его схемному изображению и далее войдя в Help или из литературы Наши же инструкции по использованию приборов будут даваться в соответствующем контексте по мере необходимости их конкретного применения

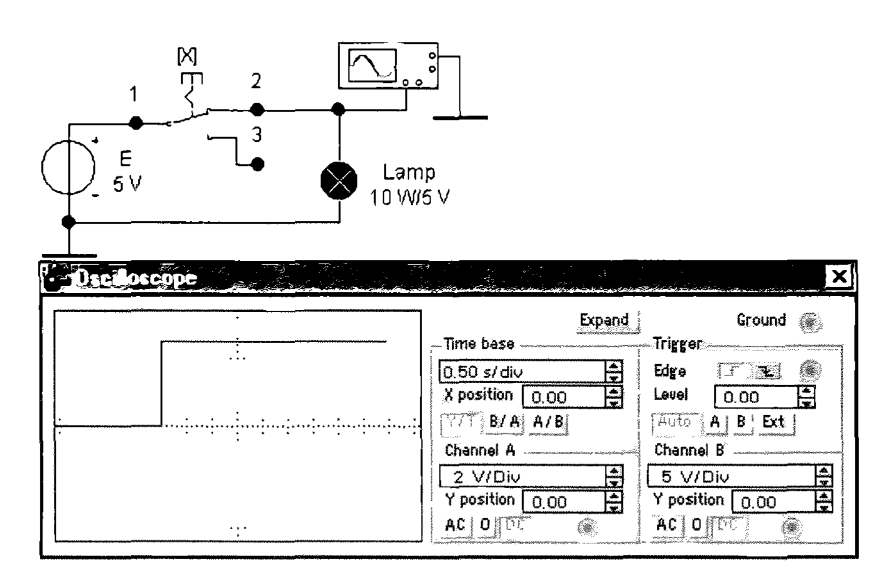


Рис 8 Наблюдение поведения модели на осциллоскопе (**EWB**)

Совет **(EWB)** Не пытайтесь включить в схему развернутое изображение осциллоскопа, так как перечисленные выше клеммы (входы каналов. заземление (Ground) и синхронизация) на этом изображении не активны, но если в схеме не соединены необходимые клеммы свернутого изображения виртуального схемного осциллоскопа, то это равноценно не включенному реальному прибору.

После установок режимов осциллоскопа обязательно надо щелкнуть ЛКМ на свободном поле, иначе программа будет находиться в режиме ожидания дальнейших команд для осциллоскопа, которые допускается выполнять и в процессе его работы, но клавиша переключателя (Switch) при этом не будет управляться с клавиатуры.

Соединим вход канала А с верхним выводом лампы и для определенности отсчета уровней заземлим осциллоскоп (рис. 8). Увеличим чувствительность вертикальной развертки канала А для более полного использования экрана, установив ее равной 2 V/Div (Вольта на деление) Остальные предустановки принимаем по умолчанию Щелкаем ЛКМ на свободном поле и переходим к моделированию работы схемы

Для наблюдения картины на экране осциллоскопа устанавливаем в виртуальном выключателе Activate simulation (включение моделирования) , указатель на I (In – включено) и делаем щелчок ЛКМ Нажимая на клавишу [X], наблюдаем за соответствием положения ключа и разворачивающегося луча на осциллоскопе, а также горением лампы высокий уровень совпадает с замкнутым ключом и горящей лампой, низкий – с разомкнутым ключом и не горящей лампой (рис 8) Выйдем из режима анализа поведения модели, нажав ЛКМ на O (Out)

Совет **(EWB)** Не забывайте останавливать режим анализа при переходе к другим этапам работы с программой, в противном случае процесс счета продолжается, о чем свидетельствуют «бегущие» цифры в окошке указателя текущего времени счета, расположенного в нижней информационной строке текущих команд и состояний Если результат получен, то далее только «забивается» оперативная память

Изменим полярность сигнала (питания лампы) Сделаем вначале копию исходной цепи Для этого производим ее выделение, нажимая на ЛКМ и обводя ее выделяющей рамкой, либо из меню Edit>Select All и далее, пользуясь стандартными командами Windows – Copy и Past Здесь надо иметь в виду, что при таком копировании не следует включать вовнутрь выделения схемный осциллоскоп его копирование в пределах одного файла не допускается (Если требуется полная копия, то надо сохранить этот файл под новым именем и работать с этим новым файлом)

После того как копия возникнет на экране, наводим курсор на любой ее активный (выделенный красным цветом элемент) указатель – он превратится в изображение руки, нажмем ЛКМ и буксируем схему вниз При достижении нужного положения (схемы не должны пересекаться) надо отпустить ЛКМ Привычная операция отмены (Undo) здесь, увы, отсутствует Правда, в меню File есть опция Revert Save (возврат к сохраненному), но она требует постоянного сохранения редактируемых схем, а это не всегда удобно Проще перед вставкой копии «прокрутить» экран, щелкнуть ЛКМ на свободном поле и только после этого делать вставку Кроме того, в программе EWB проводники, соединяющие схемные компоненты, выполнены свободно растяжимыми и сжи-

маемыми при закрепленных концах Этим можно пользоваться для придания графике схем нужной конфигурации

Совет (**EWB**) Обязательно освобождайте центральную часть экрана для получения копии, а также буксируйте схему на новое место сразу после копирования (пока она активная – «красная»), иначе, особенно в сложных схемах, повторно выделить наложенные друг на друга схемы и «растащить» их практически не удастся

На экране имеются две схемы Обозначим в них лампы Lamp1 и Lamp2, а батареи E1 и E2, соответственно, и во второй (правой) схеме отсоединим батарею от цепи

Совет (**EWB**) Для устранения какого-нибудь соединения достаточно выделить любую его конечную точку, отвести ее вместе с проводником в свободное место и отпустить ЛКМ, т е как бы один конец провода бросить в пустоту Можно также выделить проводник однократным нажатием ЛКМ и затем удалить его любым стандартным приемом, например, воспользовавшись клавишей «Delete» Аналогично удаляется любой ненужный компонент или даже часть схемы, но после ее стандартного рамочного выделения

Затем выделим батарею E2 и, воспользовавшись кнопками вращения, повернем ее вокруг вертикальной оси, а далее вывод «+» батареи соединим с землей, а «–» с переключателем (рис 9)

Кроме того, соединим верхний вывод лампы 2 с клеммой входа канала B схемного осциллоскопа Для того чтобы лучи не совпадали на экране, сместим их нулевые линии развертки (Y position) для луча A вверх на 0 20 и для B вниз на –0 20 Можно также раскрасить лучи в разные цвета, задав соответствующие цвета подводящим проводам

Включив процедуру анализа клавишей In и включая [X], пронаблюдаем за состояниями в схеме (рис 9) Видно, что при отрицательном питании ничего в логике работы не изменяется, если только договориться высоту уровня отсчитывать по модулю напряжения (возможны и иные конвенции) Тогда нижняя часть луча B, соответствующая физически напряжению –5 V (положительный вывод батареи E2 заземлен и имеет потенциал 0 V), имеет по принятому условию «высокий» уровень, равный по модулю 5 V

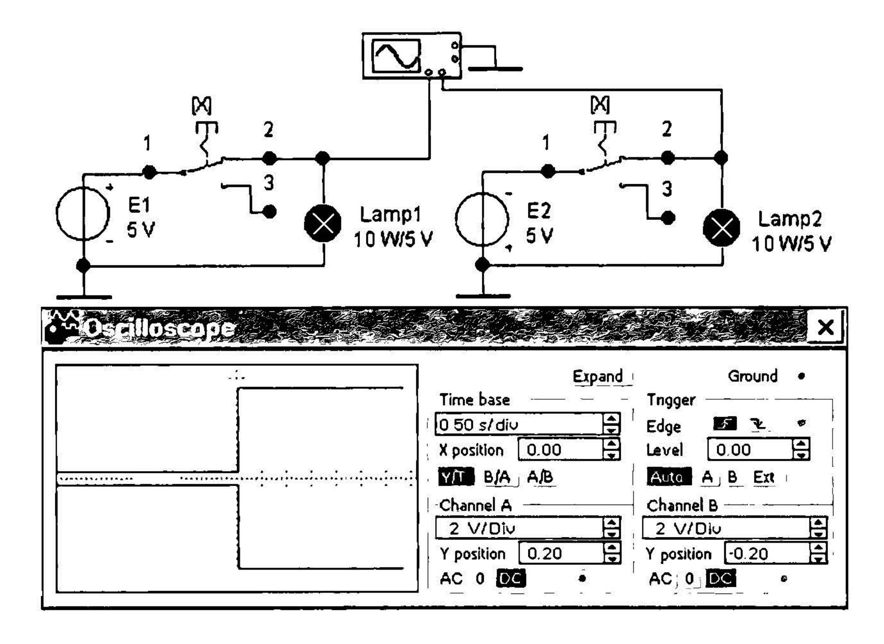


Рис 9 Наблюдение поведения модели на осциллоскопе при положительном и отрицательном сигнале (**EWB**)

В, соответствующая физически напряжению –5 V (положительный вывод батареи E2 заземлен и имеет потенциал 0 V), имеет по принятому условию «высокий» уровень, равный по модулю 5 V

Введем далее следующее соглашение о логических уровнях для рассматриваемой *положительной логики* верхнему уровню соответствует *логическая единица* (лог. 1), а нижнему – *логический нуль* (лог 0) При условии лог 1 – ключ замкнут (это истинное значение состояния в положительной логике независимо от знака сигнала). Похожая ситуация возникает при чтении напечатанного «черным по белому» книжного текста или с экрана монитора черные буквы на листе бумаги не отражают, а на мониторе не излучают свет – отрицательный сигнал Проведя стандартное выделение текста в редакторе Word, получим белые буквы на черном фоне – положительный сигнал – свет теперь излучается выделенными буквами. Однако в обоих случаях действует своеобразная положительная логика

Разработка модели в программе Micro-Cap

Проведем моделирование работы простейшей цепи в программе **MC**

Открываем основное рабочее окно программы (рис 10) и в нем нажимаем на кнопку New (новый) или с клавиатуры Control+N Затем, в появившемся ниспадающем меню, выбираем опцию схемы – Schematic (рис 11).

Наводим курсор на кнопку с изображением стрелки - Select Mode (выбор объекта) и нажимаем ЛКМ Подводим стрелку-курсор к надписи Component (выбор компонента) и далее в открывшемся меню выбираем Animation (анимация), а затем Digital Switch (цифровой ключ) После щелчка ЛКМ переводим курсор на свободное поле и там, после еще одного щелчка ЛКМ, получаем изображение ключа (рис 10) Затем повторяем Component>Animation и выбираем LED (Light-Emitting Diode – светоизлучающий диод – СИД), имеющий изображение в форме квадрата с выводом для его присоединения (см рис 11)

Совет (**MC**) Не забывайте каждый раз после выбора и редактирования очередного компонента переключаться в режим Select Mode (этого не надо делать только в одном случае следующая процедура заключается в выборе однотипного с предыдущим компонента). Если этого не сделать, то можно все рабочее поле покрыть одинаковыми компонентами, от которых трудно «отвязаться». Правда, в отличие от **EWB**, в программе **MC** есть операция Undo (отмена), позволяющая вносить исправления на ходу Кроме того, этот режим можно изменить, войдя в меню Options (опции) общих настроек программы Preferences (предпочтения), но это лучше выполнять, уже привыкнув к программе

В данном случае этих двух схемных компонентов достаточно для того, чтобы смоделировать логику работы лампы Остается их только соединить проводниками Сборка схем и их графическое редактирование в программе **MC** имеет свои особенности

Нажимаем курсором на пиктограмму Orthogonal Wire (ортогональные проводники) и соединяем проводником соответствующие выводы компонентов Возвращаемся в режим Select Mode.

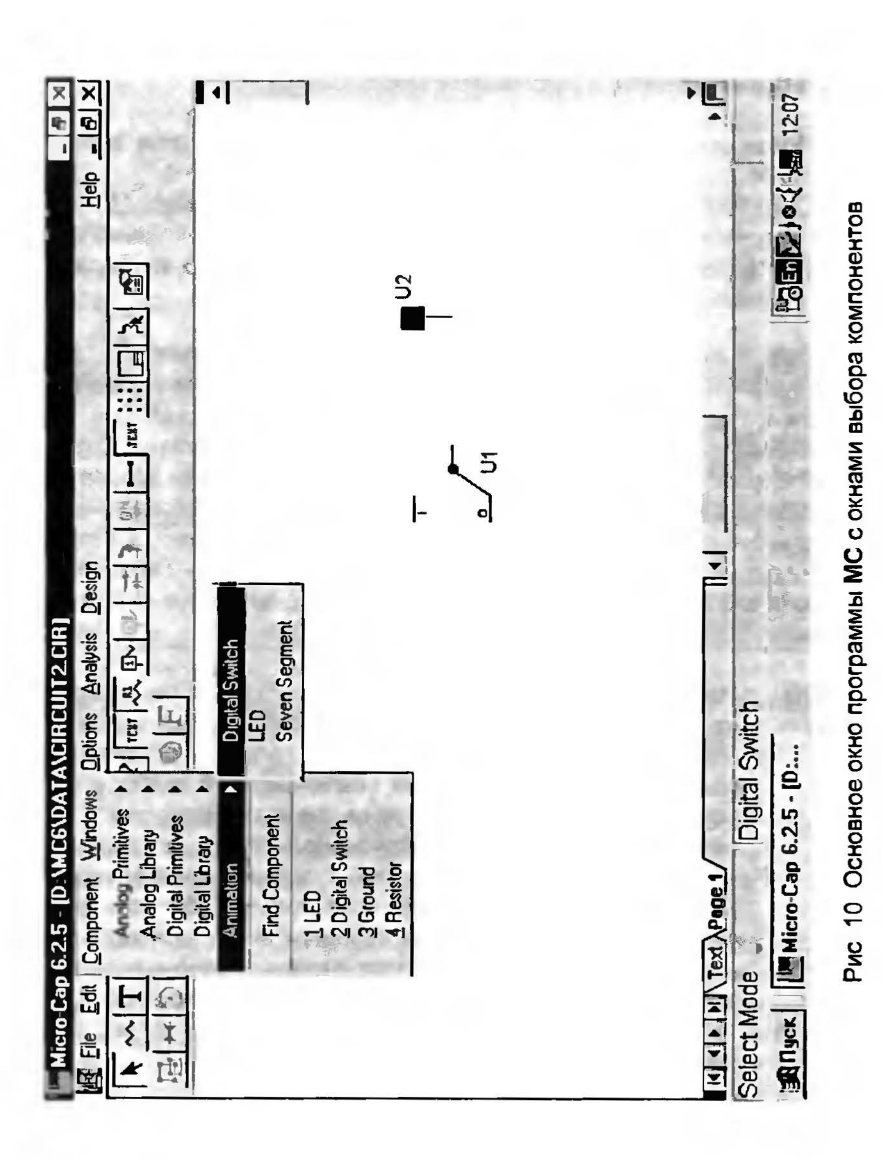


Рис 10 Основное окно программы МС с окнами выбора компонентов

Для проверки целостности цепи есть несколько вариантов

Можно нажать на пиктограмму . Node Numbers (нумерация узлов): на схеме возникнут номера узлов (рис 12,а)

Отсутствие необходимого соединения сразу обнаруживается за счет появления «лишних» узлов

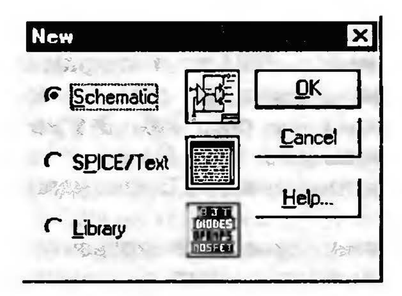


Рис 11 Окно выбора режима работы в программе MC

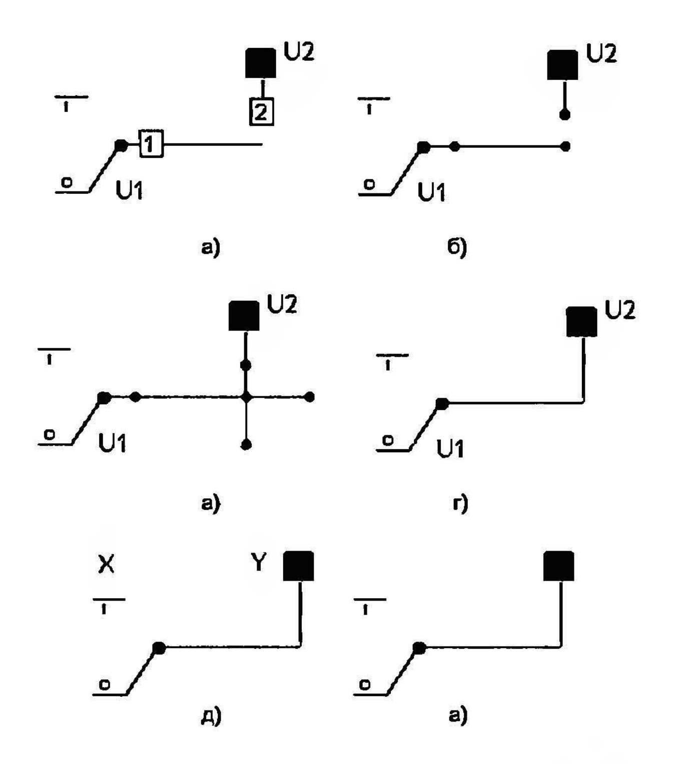


Рис 12. Графическое редактирование схем в программе MC

Можно также нажать ЛКМ по пиктограмме Pin Connections (соединительные выводы), тогда выводы будут отмечены зачерненными точками (см рис 12,б) Эта же команда позволяет «увидеть» все выводы (рис 12,в) После этого лишний проводник надо выделить, а затем, нажав на Delete, удалить (рис 12,г)

Совет **(МС)** Обязательно проводите проверку целостности схем перед тем, как проводить анализ их работы, иначе аналоговые схемы будут неработоспособны, а цифровые давать произвольные результаты

Переименуем текстовые метки (Attribute Text) позиционного обозначения компонентов U1 и U2 на X и Y, соответственно Это можно сделать либо в окне свойств, вызываемом двойным щелчком ЛКМ по УГО компонента на рабочем поле (и далее, напечатав текст в Part), либо непосредственно в самой метке Заодно метки можно буксировать в другие места (рис 12,д) Повторное нажатие на пиктограммы убирает соответствующие надписи с экрана (рис 12,е)

Совет **(МС)** Если необходимо отыскать какой-либо новый компонент, то можно обратиться к помощи (Help Mode), а можно, не открывая файлов, воспользоваться просмоторщиком библиотечных файлов и достаточно быстро найти там нужный компонент или схему и скопировать их Если же компонент сложный, то полезно скопировать и его текстовое описание и установки анализа

Наконец, перейдем от графических экзерсисов (упражнений) к осмыслению моделирования Логический переключатель X на схеме (рис 12,д) дает возможность задать один из двух уровней лог 0 (нижнее положение наклонной клавиши-перемычки) и лог 1 (верхнее положение наклонной клавиши-перемычки) Соответственно логический индикатор Y позволяет определять эти состояния в цифровых узлах Оба компонента работают в режиме анимации только при анализе схем Приведенные «логические схемы» отличаются от обычных принципиальных, эквивалентных и т п схем в них нет привычной замкнутой цепи для тока К этому надо привыкнуть На самом деле цепь имеет другой вид, но ис-

точник и заземление не показываются Вспомним как, например, выявляются «земля» и «фаза» в обычной электрической розетке с помощью отвертки-индикатора, содержащей внутри корпуса неоновую лампу с последовательно включенным токоограничивающим резистором Вот мы (осторожно!) вставляем жало индикатора в один из выводов розетки и дотрагиваемся указательным пальцем до металлического вывода на колпачке Если индикатор исправен и есть напряжение, то он загорится при контакте с фазовым выводом Казалось бы, что здесь тоже нет замкнутой цепи Реально же при касании пальцем мы создаем замкнутый контур, в котором играем роль обкладки конденсатора замыкающего цепь индикатора на землю Не бойтесь, при соблюдении правил электробезопасности, протекающий ток составит десятые доли миллиампера В виртуальном же эксперименте нет и этого В рассматриваемой логической цепи индикатор имеет один условный вывод (как жало отвертки-индикатора) и он загорается при наличии высокого уровня напряжения Здесь низкий уровень 0 соответствует «сигнальной земле», а 1 – это высокий уровень напряжения Логика – положительная и сигналы положительные (Подобные «однопроводные» схемы можно выполнить и средствами программы **EWB**, но это будет сделано в следующих разделах)

Для проведения анализа вызываем в окне Analysis определенный тип моделирование переходных процессов (Transient на рис 13)

Соглашаемся с установками, предлагаемыми в нем по умолчанию, и даем команду на моделирование, нажимая на кнопку Run (пуск) в левом углу этого окна На экране возникнет окно анализа (Transient) с графиком (осциллограммой) состояния

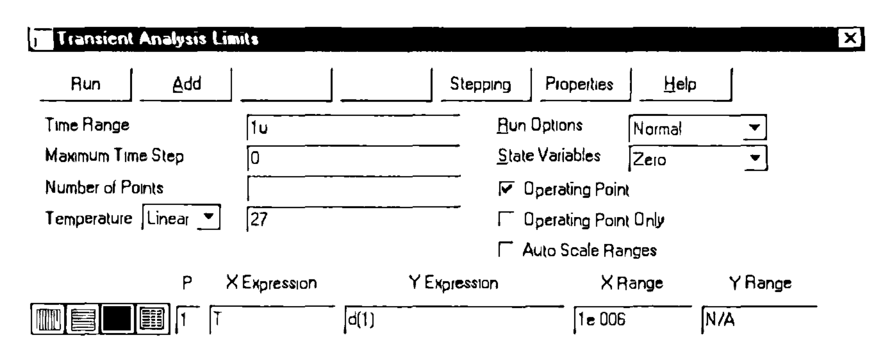


Рис 13 Окно установки параметров анализа переходных процессов в программе **МС**

цифрового узла 1 (d1) Здесь d – от digit (цифровой) Нажав также Shift+F4, можно совместить на одном экране два окна Transient и схемное (рис 14)

Логическое состояние узла на схеме получаем, «утопив» кнопку Node Voltages (узловые потенциалы) Далее, нажимая курсором ЛКМ в области переключателя, анимируем его работу по каждому щелчку его состояние изменяется, причем изменяется и уровень выходного сигнала Не следует, однако, сразу ожидать, что одновременно изменится и горение индикатора или отмеченный потенциал узла Дело в том, что в программе **МС**, в отличие от **EWB** надо заново дать команду на моделирование Run или нажать в окне Transient на кнопку ►. Можно также воспользоваться клавишей F2 В результате этих манипуляций можно пронаблюдать два возможных логических состояния простейшей схемы вход X=0, выход Y=0, СИД не горит (рис 15,а), вход X=1, выход Y=1, СИД горит (рис 15,б)

Во избежание недоразумений, заметим, что в связи с использованием черно-белой книжной графики вид свечения индикатора на экране изменен по сравнению с принятым по умолчанию

В данном тексте, как и в программе **EWB**, свечение соответствует зачернению индикатора (рис 15,б), а не горящий индикатор выполнен в виде рамки с просветом (рис 15,а)

В исходной программе не горящий индикатор – черный, а горящий – красный Подобное изменение выполняется следующей последовательностью команд выделение (рамочное или Edit>Select All), Edit>Change>Properties>Digital 0>Color (белый) Digital 1>Color (черный)

Показания логических состояний узлов 0 или 1 и, соответственно, вид свечения имеют смысл только при включенном окне Transient

Совет **(МС)** Не пытайтесь произвести какие-либо соединения с горизонтальными линиями уровней 0 и 1 на УГО переключателя Digital Switch – это не выводы здесь только один вывод – от цифрового узла Необходимо также иметь в виду, что если не закрыто окно анализа, то большинство команд вне этого окна программа не воспринимает и действует как стояночный тормоз на автомобиле «Не дергайтесь понапрасну!»

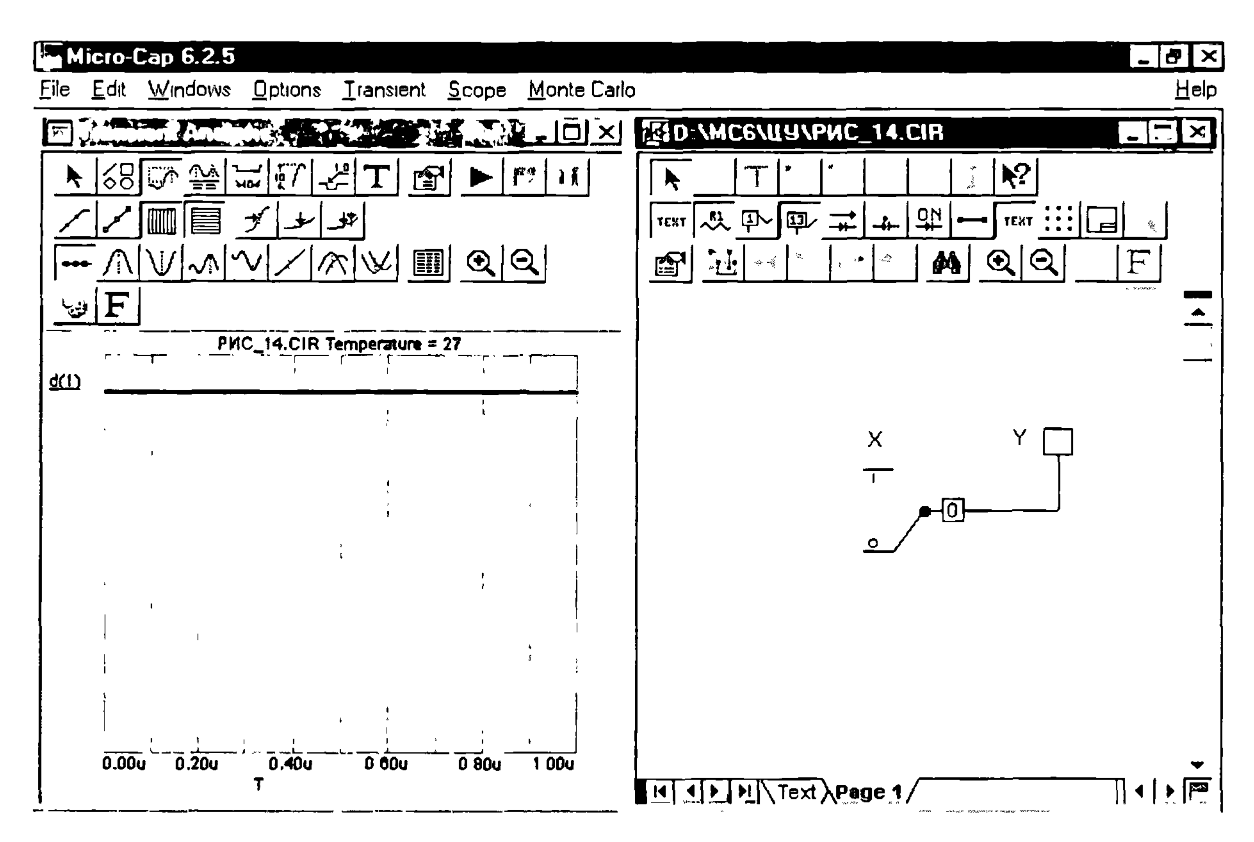


Рис 14 Окно наблюдения переходных процессов (MC)

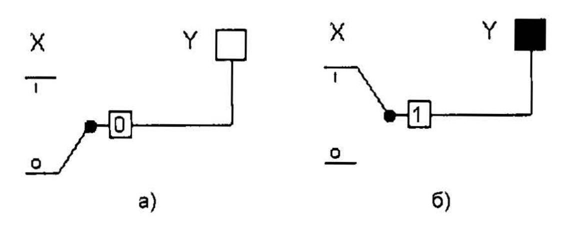


Рис 15 Логические состояния простейшей схемы (**МС**)

Задача об охранной сигнализации

Прежде чем двигаться дальше, решим элементарную задачу на сообразительность Пусть необходимо создать простейшую систему охранной сигнализации, состоящую из источника питания, сигнальной лампы и провода, огораживающего объект от несанкционированного доступа В реальной задаче разрыв провода, например наклеенного на лобовое стекло автомашины или разрыв контактных шлейфов в дверных проемах должен привести к появлению светового сигнала

Казалось бы, здесь нет ничего сложного берем ту же схему (рис 7), в которой ключом моделируем разрыв охранного провода Вот только одна «закавыка» при *размыкании* контакта лампа должна не гаснуть, а загораться! Абсурд? Нет, решение, и крайне простое, хотя и оригинальное, существует Мы предлагаем вначале попытаться найти его самостоятельно, отложив книжку в сторону Нахождение решения, как открытие, принесет громадное удовлетворение Не зря ведь по преданию, совершив открытие, Архимед голым выскочил из ванной, а Пифагор на радостях отдал на заклание музам 100 быков Тут, конечно, случай поскромнее И все-таки, и все-таки

Посоветовав читателям находить решение эвристически, применим следующий метод Поскольку искомая цепь является как бы зеркальной по отношению к цепи обычной лампы, то найдем ее по известным правилам, основанным на следующих свойствах схем Проводники, соединяющие на схемах двухполюсные компоненты, принимаются идеальными, т е сверхпроводящими, в отличие от всей остальной, не занятой компонентами плоскости чертежа схемы, которая тоже предполагается идеальной, но изолятором В воображаемом зеркальном пространстве, грубо говоря, надо поменять проводники и изоляторы Для этого на плоско-

сти чертежа внутри и вне контура схемы поставим по узлу (a и b) и соединим их между собой ветвями (пунктир) с однократным пересечением по всем двухполюсникам (рис. 16,а).

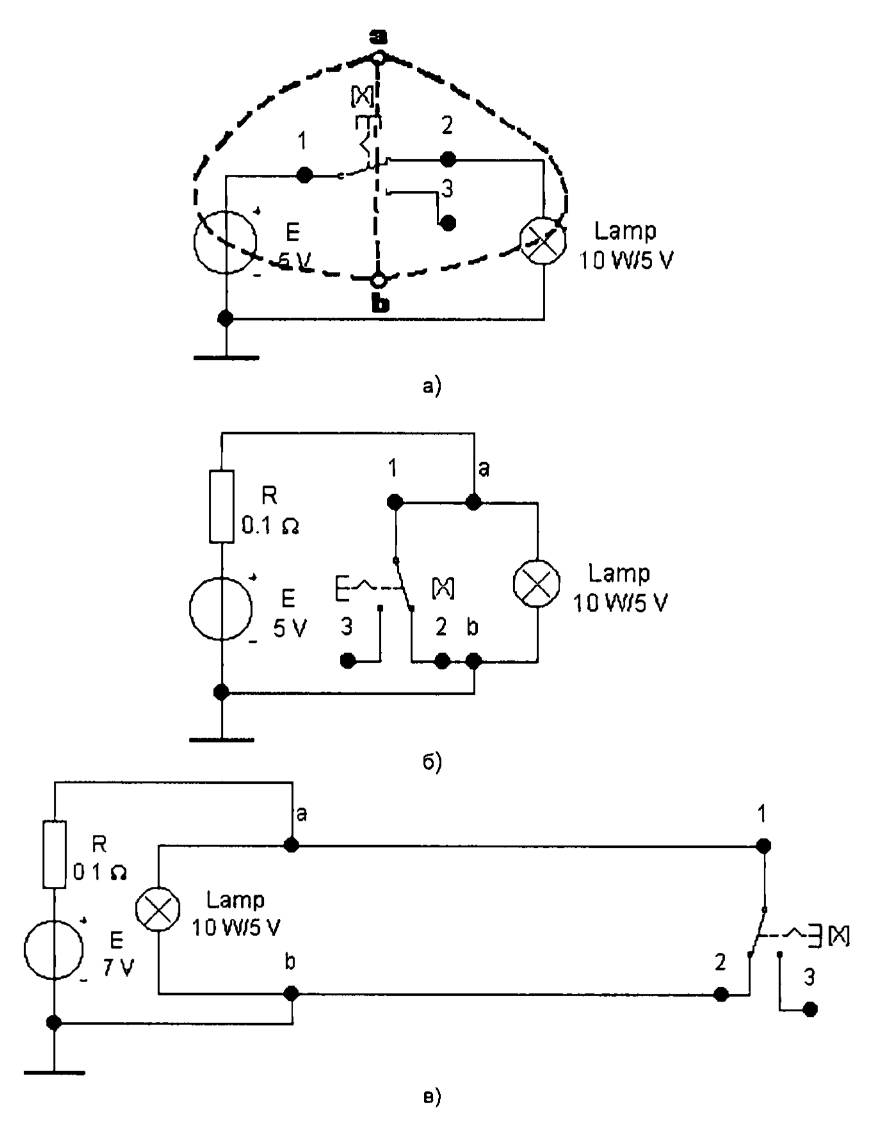


Рис. 16. Решение задачи об охранной сигнализации (**EWB**)

Во вновь полученной схеме заменим двухполюсники аналогичными, но с обратными характеристиками Напомним, что подобным приемом в аналоговых цепях из последовательного резонансного контура получается параллельный В данном случае идеальный разомкнутый контакт заменится замкнутым (и, наоборот) и разместится не последовательно, а *параллельно лампе* Только и всего При замкнутом ключе лампа «закорочена» и не горит, а при разрыве она загорается При реализации такой логически правильной схемы в бытовой электросети будет осуществлен режим короткого замыкания источника питания (авария!) Мы же находимся в виртуальном пространстве, где подобные проблемы отсутствуют Переход к реальным цепям требует включения не только реальных источников, но и головы· перед тем как что-то делать, надо подумать о возможных последствиях Компьютерное моделирование как раз и позволяет их «увидеть»

Впрочем, при педантичном использовании описываемого приема, следовало бы заменить идеальный источник постоянного напряжения с ЭДС 5 В на идеальный источник тока 2 А, и тогда схема осталась бы работоспособной Для данной задачи схему можно просто слегка подправить, введя последовательно с батареей небольшой балластный резистор R=0 1 Ом В реальном устройстве пришлось бы поднять ЭДС и соответственно выбрать сопротивление с большим номиналом, но все это не существенно для логики Окончательно получаем схему, показанную на рис 16,б Замыкая и размыкая ключ в этой схеме, убеждаемся, что она отвечает поставленной задаче На рис. 16,в предыдущая схема просто перечерчена так, что источник питания и сигнальная лампа для наглядности локализованы в левой части экрана, а охранная петля a-1-2-b вынесена далеко вправо Лампа не горит до тех пор, пока нет разрыва а любом месте этой петли

Проведенную операцию перехода от схемы с положительной логикой к схеме с отрицательной логикой назовем *инверсией*

В этом примере мы встретились со случаем использования отрицательной логики И дело здесь не в полярности сигнала смена полярности на обратную, не изменит логики работы схемы

За *отрицательную логику* в электрических цепях примем правило обратное, сформулированному выше для положительной логики, а именно верхнему (по модулю) уровню соответствует лог 0, а нижнему – лог 1 При условии лог 1 – ключ будет замкнут (это истинное значение состояния независимо от знака сигнала), а сигнал будет иметь низкий уровень. В отрицательной логике все как бы «вверх ногами»

Вернемся к рассмотренному примеру (рис 16,б) Здесь «истину» отображает замкнутый ключ, соответственно логическая 1 (лог 1) и низкий (по модулю) уровень напряжения (здесь 0 Вольт) Разомкнутый же ключ отображает «ложь», логический, но не физический 0, (не «Вольты» равны нулю, а состояние 0) Соответственно уровень напряжения будет высоким по модулю (здесь 5 В)

Рассмотренные два варианта исчерпывают все возможные случаи логических схем с одним ключом Со схемотехнической точки зрения логическая часть схемы представляет собой одноэлементный (вырожденный) четырехполюсник, на входе которого подсоединена батарея, а на выходе – лампа Поскольку ключ является двухполюсным элементом цепи, то существуют только два различных способа его включения в последовательную ветвь или в параллельную (развороты ключа на 180°, перестановки элементов в одной ветви, другое начертание параллельных ветвей и т п не приводят к новым логическим вариантам) Возможности логики существенно меняются, если в схему ввести два ключа

Введем в начальную схему лампы (рис 7) последовательно еще один ключ и выберем для управления ими клавиши A и B (рис 17,а), выход обозначим через Y (позиционное обозначение источника E и параметры лампы не отображены на экране).

Совет **(EWB)**	Обязательно задавайте разные имена управляющим клавишам для обеспечения независимой работы нескольких ключей в одной схеме

Припишем буквам A, B, Y смысл булевых переменных, отражающих логическое состояние контактной схемы, тогда каждая из них может принять только два значения лог 0 и лог 1 Однако, состояниями A и B мы можем распорядится произвольно – это независимые переменные, состояние же выхода Y будет функцией от них Итак, Y является функцией двух переменных A и B, каждая из которых может принять независимо два значения: лог 0 и лог 1. Перебор всех возможных пар значений A и B дает четыре случая лог 0, лог 0, лог. 0, лог. 1; лог 1, лог. 0, лог. 1, лог. 1.

Экспериментально зададим эти варианты, управляя ключами A и B, регистрируя состояние выхода Y Результаты будем заносить в специальную таблицу в виде логических 0 и 1, помня, что 1 соответствует здесь замкнутому ключу, а 0 – разомкнутому (слово «лог » опустим для краткости). Подобные таблицы называют таб-

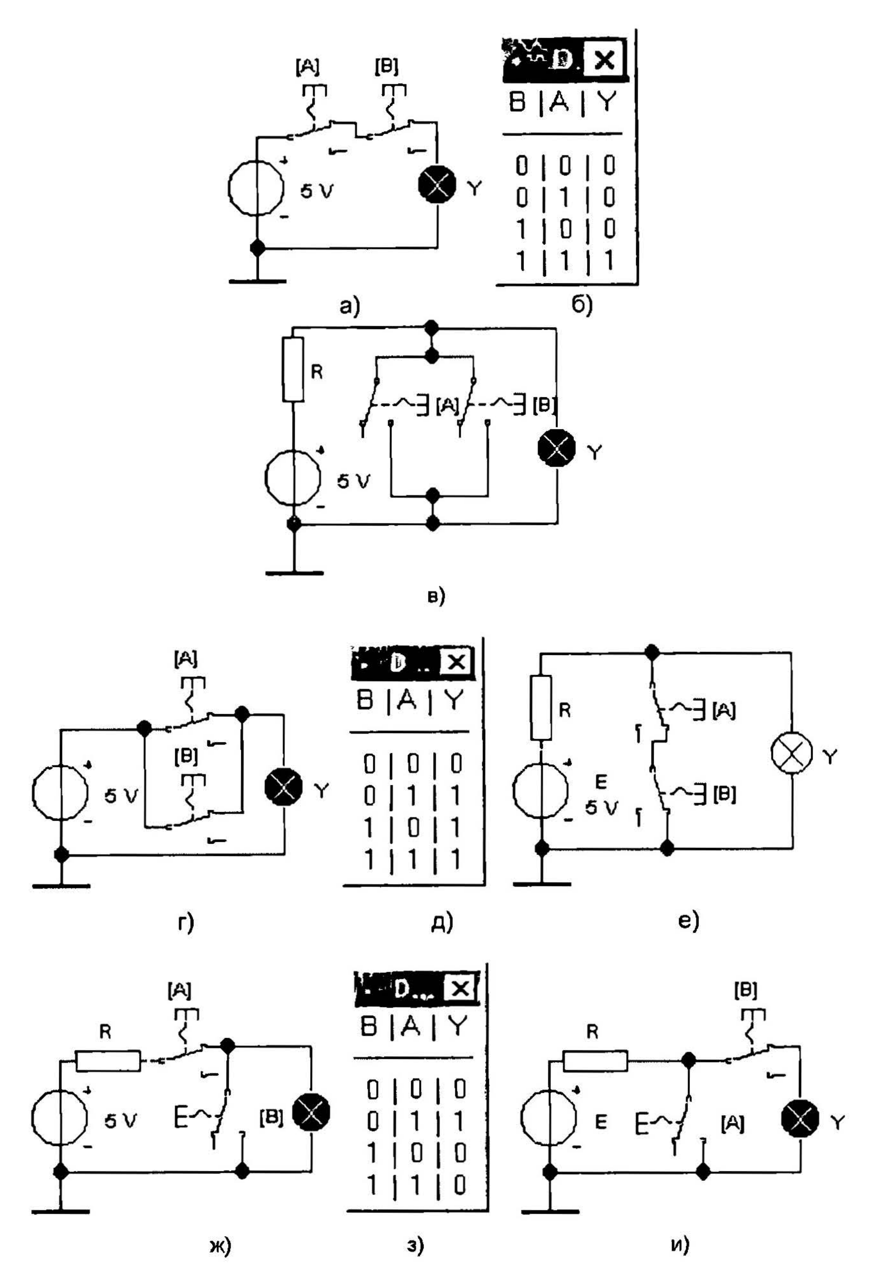


Рис 17. Контактные схемы с двумя ключами (**EWB**)

лицами истинности (ТИ) Запись состояний в таблице выполняем упорядоченно строки боковика представляют разряды двоичных чисел ВА Таким образом, А – младший значащий разряд, В – старший значащий разряд числа Тогда числа с указанием в нижнем подстрочном индексе системы счисления (2 - двоичная, 10 – десятичная) сверху вниз имеют следующие значения $(00)_2=(0)_{10}$; $(01)_2=(1)_{10}$, $(10)_2=(2)_{10}$, $(11)_2=(3)_{10}$ Всего здесь возможны четыре состояния

Для печатания таблицы истинности нажмем ЛКМ по надписи Window в верхней строке рабочего окна (меню команд) и далее Description (описание) или сразу с клавиатуры Ctrl+D Поместив курсор в появившуюся текстовую область, печатаем таблицу, заполняя ее результатами виртуального эксперимента (рис. 17,б)

Из эксперимента видно, что логический элемент из последовательных ключей дает лог 1 на своем выходе только в одном случае, когда оба ключа А *И* В имеют значение лог 1. Логической операцией, которую выполняют в данном случае ключи, является умножение аргументов, т.е. Y=AB Подобная операция и логический элемент называется *И*, соответственно, на английском AND

Инвертируем эту схему аналогично описанному выше для схемы на рис 16,а В схеме образуется еще одна параллельная ветвь с ключом Соберем теперь новую схему, соединив ключи параллельно (рис. 17,в) Повторив эксперимент аналогично предыдущему, видим, что в отрицательной логике в таблице истинности три нуля появятся на месте прежних трех единиц в нижней строчке (рис 17,б) Этот случай будет соответствовать здесь двум разомкнутым ключам (А=лог 0, В=лог 0) и Y=лог 0 – высокий уровень в отрицательной логике лампа горит

Далее от схемы, показанной на рис 17,а, перейдем к схеме с параллельным соединением ключей, но образующим вместе последовательное соединение батареи, этой «связки» ключей и лампы (рис 17,г) Составим таблицу истинности (рис 17,д)

Для этой схемы лог 1 на выходе получается если хотя бы один ключ А *ИЛИ* В будет иметь значение лог 1 (замкнут) Логической операцией, которую выполняют в данном случае ключи, является сложение аргументов, т е Y=A+B Подобная операция и логический элемент называется *ИЛИ*, соответственно на английском OR (Обратите внимание на отличие сложения логических единиц здесь 1+1=1)

Отметим еще некоторые особенности исследуемых схем Схемы и логические операции на рис 17,а и рис 17,г двойственны (двойственны между собой и схемы на рис 17,в и рис 17,е)

При инвертировании схемы *И* положительной логики (рис 17,а), получаем в отрицательной логике (рис 17,в) для инверсных переменных, отмеченных штрихами, Y`= A`+B`, что соответствует операции *ИЛИ* Этот результат виден и в таблице (рис 17,б), достаточно только в ней все 0 поменять на 1, и наоборот

Аналогично, при инвертировании схемы *ИЛИ* положительной логики (рис 17,г), получаем в отрицательной логике (рис 17,е) для инверсных переменных, отмеченных штрихами, Y`= A`B`, что соответствует операции *И* Здесь в таблице (рис 17,д) будут три единицы на месте прежних нулей, и только в этом случае лампа не горит, так как в отрицательной логике 1 – это низкий уровень

Проведенные сопоставления позволяют сделать определенные выводы относительно применения двойственных и инвертированных функций и схем, в зависимости от типов используемой логики (положительной или отрицательной)

Рассмотренные схемы не исчерпывают всех возможных случаев с двумя ключами Используя как бы смешанное соединение, приходим к схеме, показанной на рис 17,ж Таблица истинности для нее показана на рис 17,з Логической функцией выхода будет Y=AB`, т е выход равен 1 только, когда ключ А замкнут, при разомкнутом В Функция Y повторяет сигнал, подаваемый на вход А, если нет «запрета» по входу В Инвертирование этой схемы дает последний вариант, представленный на рис 17,и В данном случае, в отличие от предыдущих (рис 17,а, в, г, е), важно, при сравнении, соответствие наименований ключей их положению в схеме

Не углубляясь далее в логику контактных схем, перейдем к другой элементной базе, а именно к логическим элементам (ЛЭ), выполненным на диодах, транзисторах и интегральных микросхемах

1.2. Компоненты цифровой электроники

«А теперь самое время попытаться найти ключ к разгадка асах историй», – сказал Шалтай-Болтай

Р М. Смаллиан. Алиса в Стране Смекалки

Дискретные логические компоненты

Диодно-резистивная логика

Идеальный диод представляет собой полупроводниковый прибор, имеющий два вывода (двухполюсник) Фактически это готовый ключ, но не имеющий механических контактов Состояние диодного ключа зависит от знака приложенного к нему напряжения его сопротивление при прямом включении близко к нулю, а при обратном – близко к бесконечности Подача на диод соответствующих сигналов открывает его или оставляет запертым

Пусть у нас есть два диода, соберем из них схему, аналогичную контактной схеме *ИЛИ* (рис 17,г) Для задания сигналов лог 1 и лог 0 на входы рассматриваемых ниже логических схем соберем своеобразный генератор (задатчик) четырех двоичных чисел от нуля до трех

В задатчике программы **EWB** используем два переключателя А и В, развернутые вокруг вертикальной оси, батарею и заземление (рис 18,а) В программе **МС** нужно просто взять два цифровых ключа (рис 18,б)

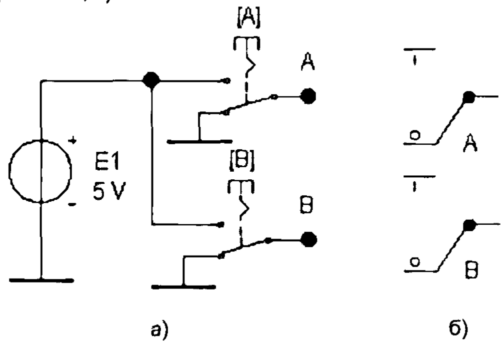


Рис 18 Задатчик двоичных чисел
а – в программе **EWB**, б – в программе **МС**

Записав этот файл, можно начинать с него сборку многих последующих схем, вызывая его и сохраняя под новым именем Можно также обойтись и копированием этой части схем

Совет При систематической работе с программами целесообразно помимо прилагаемых к ним каталогов схемных файлов завести упорядоченные библиотеки с записями собственных схем

В программе **EWB** используем задатчик и два рабочих диода, которые «достаем» из набора Diodes (диоды), воспользовавшись пиктограммой с УГО диода. Выходной сигнал снимаем с нагрузочного резистора R1 В качестве индикатора выхода используем светоизлучающий диод (СИД), по-английски Light-Emitted Diode (LED) Этот диод находится на той же диодной панели и имеет характерное УГО. При проведении моделирования в режиме излучения света две верхние стрелки, символизирующие лучи, зачернены, в отсутствие сигнала они имеют просвет

Подобные СИД широко используются для сигнализации о работе компьютеров, принтеров, так называемых сетевых адаптеров и т п устройств

Включение диода производится в соответствии с его полярностью как и у обычного диода основная стрелка в его УГО направлена от плюса к минусу Для СИД введем метку OUT (выход) и последовательно с ним включим токоограничивающий резистор R2 Окончательно схема диодно-резистивного логического элемента *ИЛИ* приобретает вид, показанный на рис 19,а

При управлении клавишами А и В получаем возможность полного перебора аргументов в этом логическом элементе и убеждаемся, что он функционирует в соответствии с таблицей состояний элемента *ИЛИ* При изменении знака сигнала на противоположный в этой схеме необходимо развернуть диоды на 180° (рис 19,б)

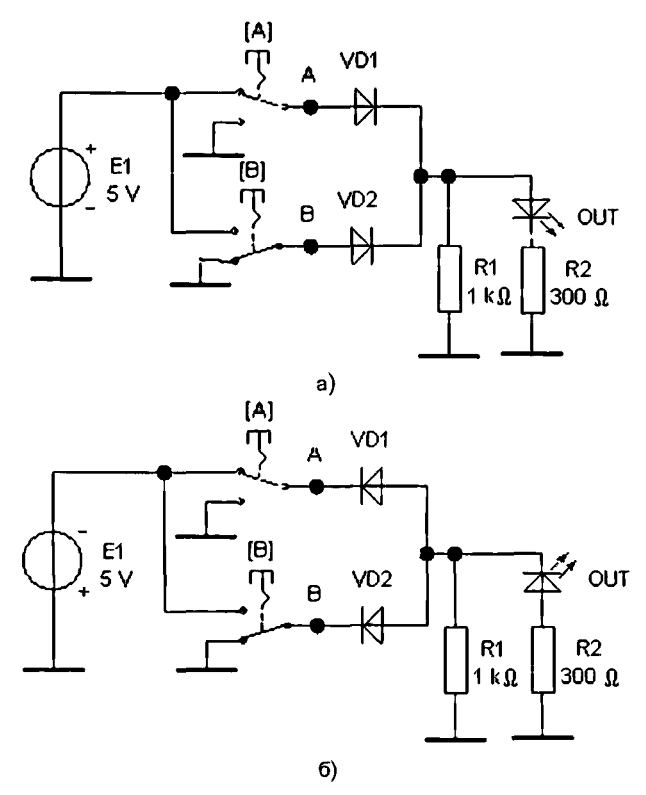


Рис 19 Диодно-резистивный ЛЭ
а – при положительном, б – при отрицательном сигнале (**EWB**)

Совет **(EWB)** При сборке схем обратите внимание на полярности включения источников, рабочих и светоизлучающих диодов, а также на номиналы использованных резисторов Если схемы не работают правильно, то проверьте все соединения, среди них могут оказаться и «ложные», в которых выводы как бы касаются узлов, но соединение отсутствует Часто подобная ситуация возникает с заземлением или выводами двух компонентов между собой Не стесняйтесь, «подергайте или пошевелите» каждый элемент виртуальной цепи (буксировкой) как при пайке реального устройства

В обоих рассмотренных случаях действует положительная логика Существенно также отметить, что роль механических переключателей здесь сильно отличается от той, которую они выполняли в контактных схемах Теперь они лишь задают необходимые входные сигналы, а собственно логическую функцию (ЛФ) «отрабатывают» диоды, не имеющие механически подвижных контактов

Соберем аналогичную схему в программе **МС** Используем задатчик 18,б Последовательно проходим по позициям Component>Analog>Primitives>Passives Primitives>Diode и в открывшемся окне выбираем первый диод, предлагаемый по умолчанию

Аналогично, последовательно пройдем по позициям Component>Analog Primitives>Passives Primitives>Resistor и Component>Analog Primitives>Passives Primitives>Ground, выбираем резистор и заземление

Совет (**МС**) В отличие от программы **EWB**, где заземление редкость и зачастую не обязательно, в аналоговой части программы **МС** хотя бы одна «земля» в схеме должна присутствовать обязательно

Затем по командам Component>Animation выбираем светоизлучающий диод LED После этого необходимо произвести графическое редактирование Для придания компонентам необходимой пространственной ориентации в **МС** предусмотрены два варианта Простые последовательные повороты на 90° осуществляются после позиционирования курсора на элементе и щелчков ПКМ при нажатой ЛКМ Повороты вокруг горизонтальной оси и вертикальной оси осуществляются после стандартного рамочного выделения элемента или их группы и выбора соответствующих команд в меню: Edit>Box>Flip X или Flip Y, здесь можно выполнить простое вращение, выбрав Rotate или с клавиатуры Ctrl+R Проведя необходимые соединения и введя необходимые позиционные обозначения элементов, получаем схему, показанную на рис 20,а

Если теперь активизировать («утопить») кнопку Node Numbers (нумерация узлов), то на схеме появятся изображения номеров аналоговых узлов в прямоугольных рамках со срезанными углами и номеров смешанных программных цифроаналоговых и аналого-цифровых узлов-интерфейсов, заключенных в прямоугольные рамки (рис 20,б) Последние отделяют

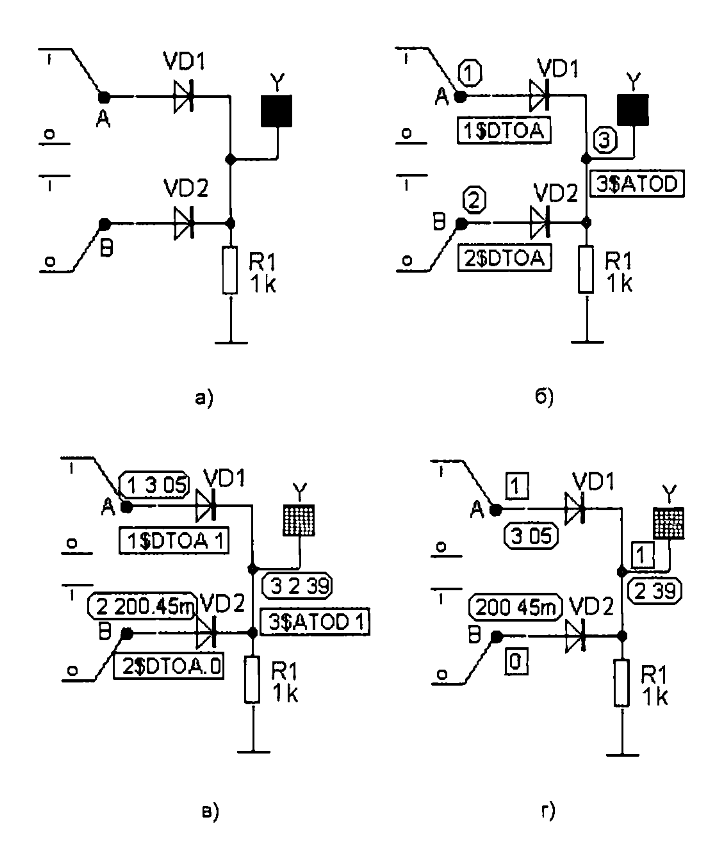


Рис 20 Диодно-резистивный ЛЭ *ИЛИ* (**МС**)

аналоговую часть схемы (в данном случае, содержащую диоды VD1, VD2 и резистор R1) от цифровой В цифро-аналоговых интерфейсах осуществляется преобразование логических уровней сигналов в аналоговые напряжения, а в аналого-цифровых интерфейсах происходит обратное преобразование Эти дополнительные программные компоненты имеют формат PSpice (Simulation Program with Integrated Circuit Emphasis – программа моделирования интегральных схем)· «имя узла интерфейса, символ $, суффикс ATOD или DTOA», где A происходит от Analog (аналоговый), D от Digital (цифровой),TO по-английски предлог «к». На наше счастье, программа выполняет процедуру присвоения автоматически и остается только принять ее к сведению Таким об-

разом, всего на схемах по команде Node Numbers могут возникнуть три типа узлов аналоговые, цифровые и смешанные

Совет **(МС)** Если метки с нумерацией узлов оказываются напечатанными друг на друга или поверх УГО схемы, то их можно отбуксировать в более удобные места или подвергнуть схему дополнительному графическому редактированию

Для выполнения моделирования даем команды Analysis> Transient>Run и в схемном окне активизируем Node Voltages для отсчета узловых потенциалов аналоговых узлов и логических состояний цифровых узлов (рис 20,в) Искомые потенциалы и состояния указываются в соответствующих рамках через двоеточие после номера узла Поскольку схема оказывается перенасыщенной информацией, то если по расположению понятно, к какому узлу она относится, то собственно нумерацию можно скрыть с экрана, повторно нажав на кнопку . Теперь мы видим (см рис 20,г) в прямоугольных рамках логические состояния 1 – для А с номером 1, 0 – для В с номером 2 и 1 – для Y с номером 3, а также потенциалы, отсчитанные относительно земли (0 V) 3 05 V для А с номером 1, 200 45 mV для В с номером 2 и 2 39 mV для Y с номером 3

Совет Не забывайте, что при наборе десятичных дробей *в рабочем поле программ* используется разделительная точка, в не запятая, например **3.05** и **200.45** Основные единицы измерения (например, V) могут не указываться, а приставки для обозначения дольных и кратных единиц измерения пишутся сокращенно, например, m – милли, k – кило

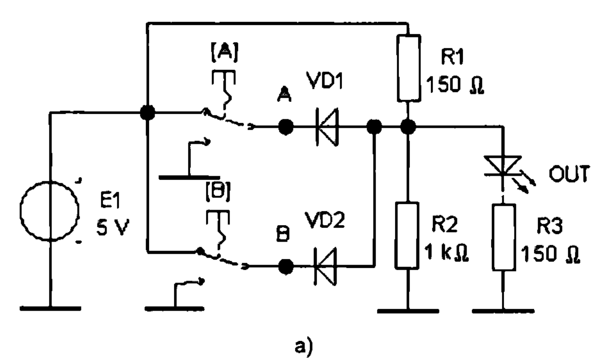
Сохраняя на экране компьютера одновременно и область (страницу) с изображением схемы (Page 1), и окно анализа (Transient Analysis) по типу, показанному на рис 14, осуществляем перебор входных логических переменных, меняя состояние переключателей А и В Каждый раз перезапуская моделирование (Run), фиксируем соответствие положений ключей и горения индикатора или значения логических состояний цифровых узлов Для схемы на рис 20 убеждаемся, что ее ТИ соответствует рис 17,д Таким образом, приходим к выводу, что это схема ЛЭ *ИЛИ*

Из проведенного анализа также видно, что уровень потенциала, принимаемого за лог 0, несколько выше 0 В, а для лог 1 ниже потенциала воображаемого источника питания Реальные микросхемы, как будет указано дальше, в зависимости от типа действительно имеют некоторые узкие зоны снизу и сверху, принимаемые соответственно за лог 0 и лог 1 Между этими зонами имеется широкая область, состояние в которой не определено

Здесь уместна некоторая аналогия с поведением электрона в атоме согласно квантовой механике он находится либо на одном энергетическом уровне (зоне), либо на другом, вопрос же о том «где находится электрон при переходе с уровня на уровень» является не корректным

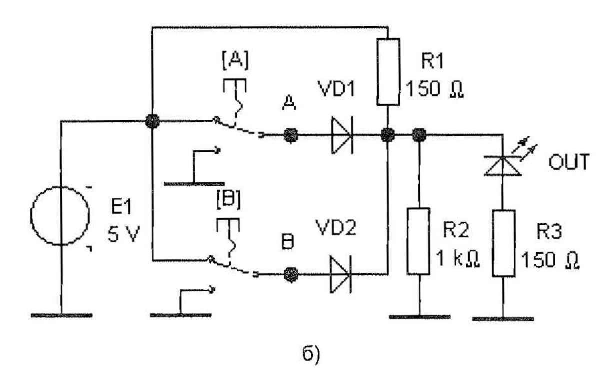
Гальваническая связь между диодами не позволяет построить схему *И,* аналогичную контактной, т е путем их последовательного соединения В подобной схеме не удастся задавать состояния диодных ключей независимо друг от друга Однако, как это следует из предыдущего, можно собрать из диодов схему *ИЛИ* для отрицательной логики (рис 11,в), а использовать ее как *И* в положительной логике Соответствующие схемы показаны на рис 21,а,б в программе **EWB** и рис 22 в программе **MC**

В последнем случае источник питания аналоговой части выбирается по пиктограмме с изображением гальванической батареи Battery или командами Component (компонент)>Analog Primitives (аналоговые компоненты)>Waveform Sources (источники сигналов)>Battery Далее двойным щелчком по УГО батареи на экране вызываем окно редактирования ее свойств и в окошке Value (значение) набираем 5, а в окошке Part (часть) E1



а)

Рис 21 (начало)



б)

Рис 21 (окончание) Диодно-резистивный ЛЭ *И*
а – при положительном, б – при отрицательном сигнале (**EWB**)

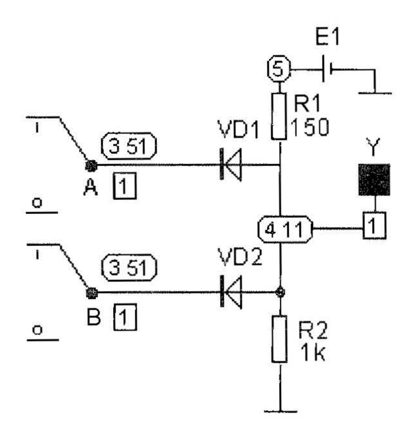


Рис 22 Диодно-резистивный ЛЭ *И* (**МС**)

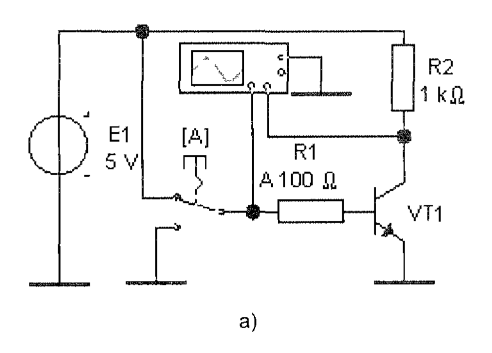
Резистивно-транзисторная логика

Использование биполярного транзистора, включенного по схеме с общим эмиттером, позволяет, подавая сигнал на базу, переводить его из закрытого состояния (большое сопротивление между коллектором и эмиттером) в открытое (насыщенное, с малым сопротивлением) В этом отношении транзистор напоминает ключ, у которого роль механической управляющей клавиши играет электрический сигнал, подаваемый на его базу Транзисторы, непосредственно соединенные в цепь, как и диоды, оказываются

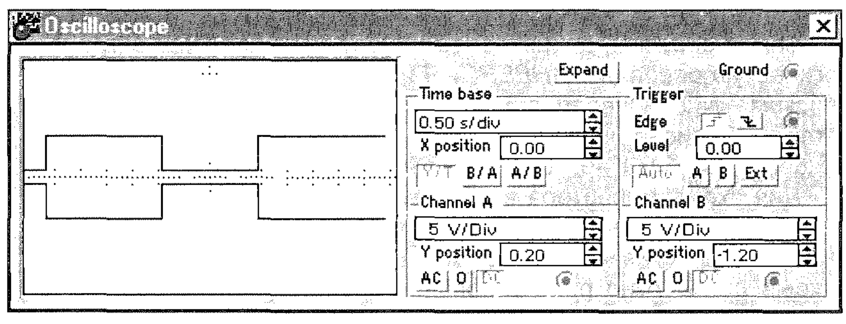
также гальванически связанными, но наличие третьего вывода позволяет так подобрать параметры схем, что они управляются независимо

Начнем с рассмотрения работы одиночного транзистора Для сборки схемы в программе **EWB** (рис 23,а) необходимо щелкнуть ЛКМ по пиктограмме с изображением транзистора и выбрать NPN транзистор для положительного сигнала

Наблюдение за работой схемы проведем с помощью осциллоскопа (рис 23,б), подав на канал А входной сигнал (верхний луч), а на канал В выходной сигнал (нижний луч), для обоих лучей на экране введено смещение по вертикали, лучи можно также дополнительно выделить цветами



а)



б)

Рис 23 Резистивно-транзисторный ЛЭ *НЕ* (**EWB**)

Совет **(EWB)** Еще раз обратите внимание на то, что осциллоскоп в программе **EWB** в схему включают, используя уменьшенное *схемное* изображение прибора, на котором есть активные выводы (рис 23,а) Показания же снимаются с виртуального изображения лицевой панели, содержащей органы управления, но на экране не связанной проводами со схемой (рис 23,б) Изображение лицевой панели вызывается двойным щелчком ЛКМ на уменьшенном схемном изображении прибора Виртуальное изображение развернутой лицевой панели можно свободно буксировать в любое удобное место экрана, в том числе поверх схемы, располагая как обычное дополнительное окно Выводы на этом изображении не активны не пытайтесь к ним что-либо подсоединить, зато активны установочные кнопки, и ими надо правильно пользоваться для работы

Из осциллограмм на рис 23,б видно, что при низком уровне входного напряжения (лог 0) сигнал на выходе имеет высокий уровень (лог 1), и, наоборот

Совет **(EWB)** Внимательно сопоставьте положение лучей на экране с масштабными коэффициентами по каналам и введенным вертикальным смещением для лучей на рис 23,б Для луча канала А (вход ЛЭ) верхнее плато соответствует +5 V (лог 1), нижнее 0 V (лог 0) Для луча канала В (выход ЛЭ) верхнее плато также соответствует +5 V (лог 1), нижнее 0 V (лог 0)

Таким образом, данный элемент осуществляет над сигналом логическую операцию *ИНВЕРСИЯ* иначе называемую *НЕ*, на английском – NOT Операция инверсии не имеет аналогии в обычной алгебре Записав инверсию в виде штриха, получаем очевидные тождества 0`=1 и 1`=0, которые и выполняет инвертор

Для схемы инвертора в программе **МС** выбираем транзистор по его пиктограмме или последовательностью команд Component >Analog Primitives>Active Devices (активные компоненты)>NPN Открыв далее окно редактирования свойств транзистора, выбираем первый предложенный из библиотечного списка и затем в окне Part вводим позиционное обозначение VT1 Результаты исследования работы инвертора показаны на рис 24,а при 0 на входе – на выходе 1, и на рис 24,б при 1 на входе – на выходе 0

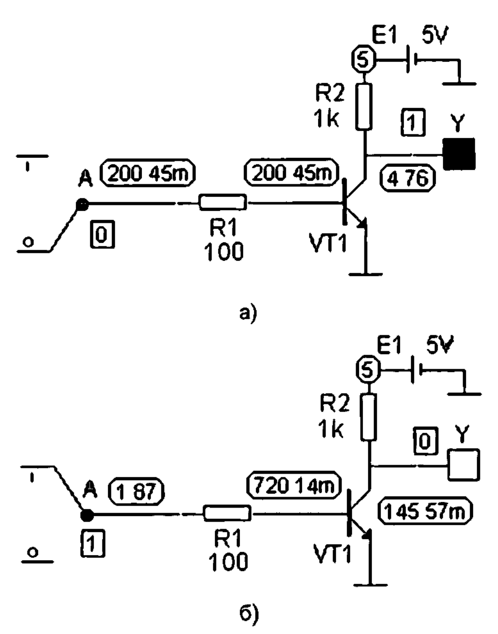
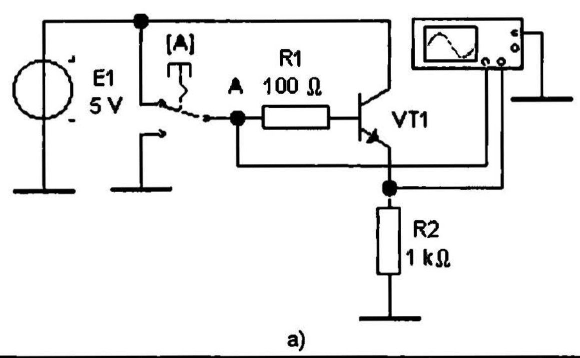


Рис 24 Резистивно-транзисторный ЛЭ *НЕ* (**МС**)

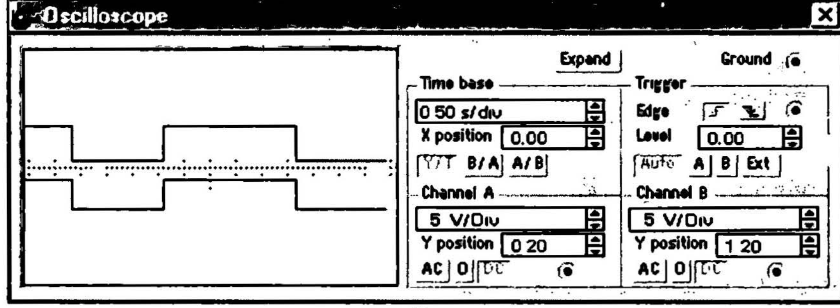
Изменяя тип транзистора (NPN и PNP) и соответственно полярность питания, можно создать еще три аналогичные схемы Однако во всех случаях, независимо от знака сигнала и типа логики, ЛЭ *НЕ* будет исправно выполнять одно и то же изменять лог 1 на лог 0, и наоборот Двукратное применение инверсии возвращает исходное значение, в чем легко убедиться, соединив последовательно два инвертора

Элемент *НЕ* является одновходовым (имеет только одну независимую переменную) и принципиально не может быть реализован в чисто контактной или диодной логике, для реализации этой функции в них пришлось бы ввести реле, трансформатор или иное устройство, способное изменять на противоположную фазу выходного сигнала по отношению к входному

Перестроив схемы ЛЭ *НЕ* так, что выходом станет эмиттер (рис. 25,а и 26), получим повторитель сигналов или буфер, служащий для согласования выходных каскадов. Из приведенной осциллограммы (рис. 25,б) или значений логических уровней на рис 26 видно, что при лог 1 на входе, на выходе также будет лог. 1, а при лог 0 на входе, на выходе также будет лог. 0



а)



б)

Рис 25 Резистивно-транзисторный буферный ЛЭ (**EWB**)

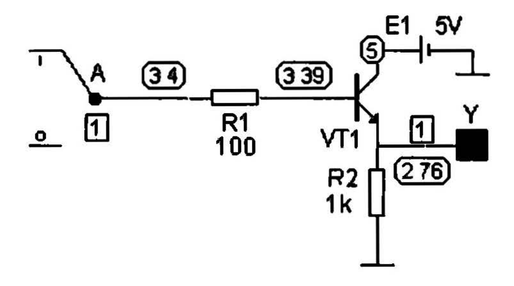


Рис 26 Резистивно-транзисторный буферный ЛЭ (**МС**)

Уникальные свойства транзистора позволяют выполнить схемы резистивно-транзисторных логических элементов *И* и *ИЛИ* аналогично соответствующим контактным схемам, показанным на рис 11,а и 11,г Проведем сборку схем аналогично предыдущим и получим схемы, показанные на рис 27, 28 в программе **EWB** и на рис 29 в программе **МС**

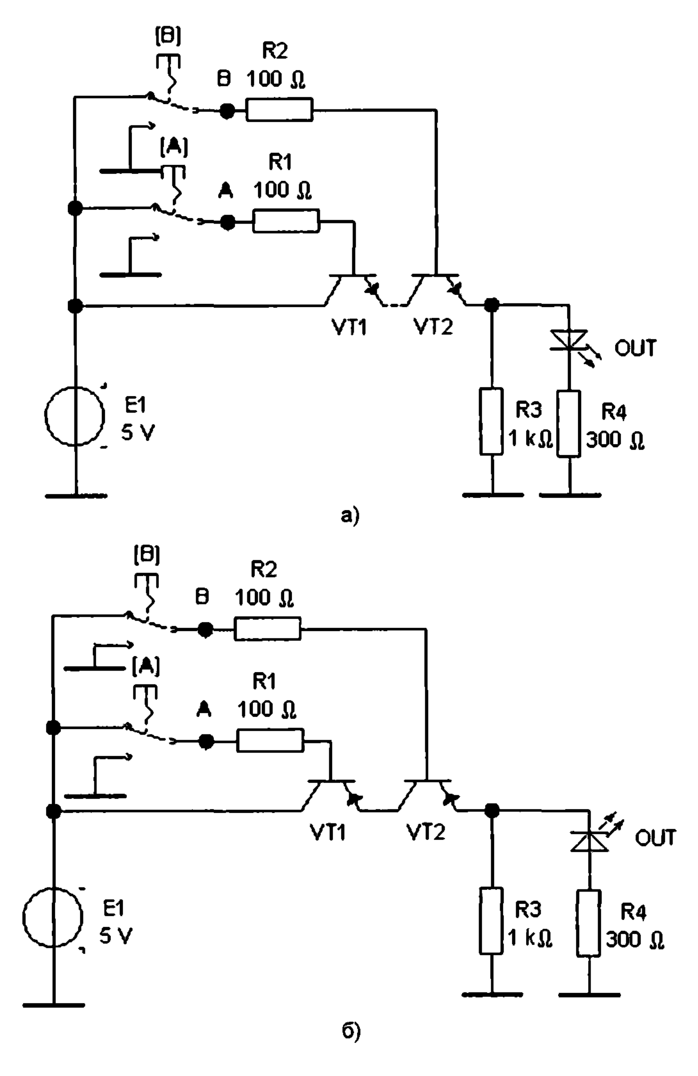


Рис 27 Резистивно-транзисторный ЛЭ *И*
а – при положительном, б – при отрицательном сигнале (**EWB**)

Совет (**EWB**) Обратите внимание на то, что при отрицательных сигналах у батареи заземлен «+», соответственно изменен тип транзистора на PNP и изменена на обратную полярность включения СИД

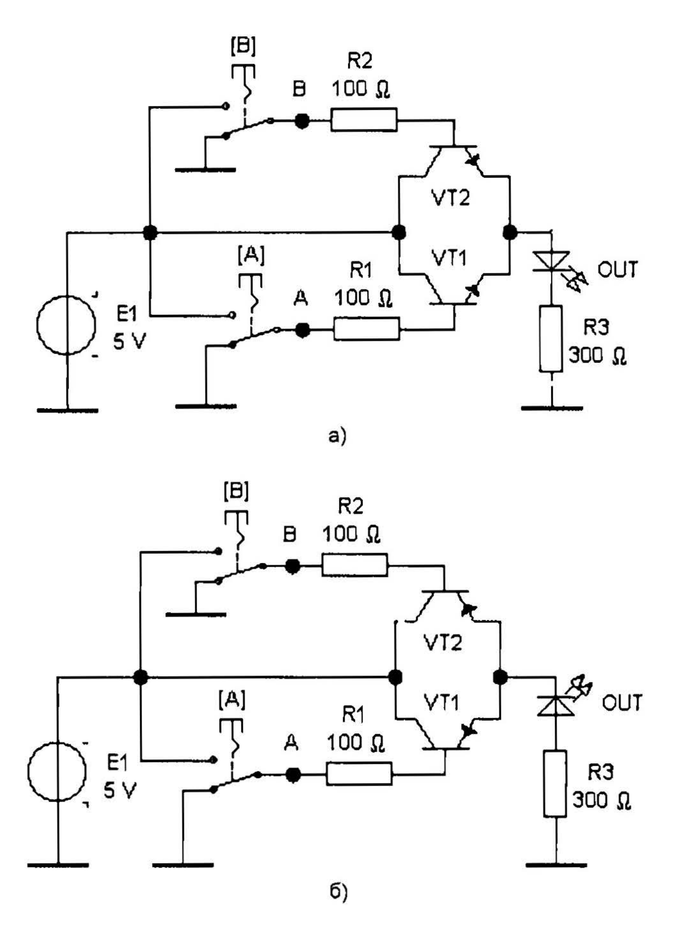


Рис 28 Резистивно-транзисторный ЛЭ *ИЛИ*
а – при положительном, б – при отрицательном сигнале (**EWB**)

Элементы *И* и *ИЛИ*, в отличие от ЛЭ *НЕ*, теоретически могут иметь любое число входов Три элемента *И, ИЛИ* и *НЕ*, образуют логически полную систему, позволяющую получить любые другие логические функции

Данный выбор не является единственным. Например, соединяя последовательно элементы *И* и *НЕ,* получим элемент *И-НЕ,* по-английски: Not AND, обозначаемый NAND. Часто этот элемент называют по фамилии американского логика XX в Генри Морриса Шеффера элементом Шеффера ЛЭ *И-НЕ* также обра-

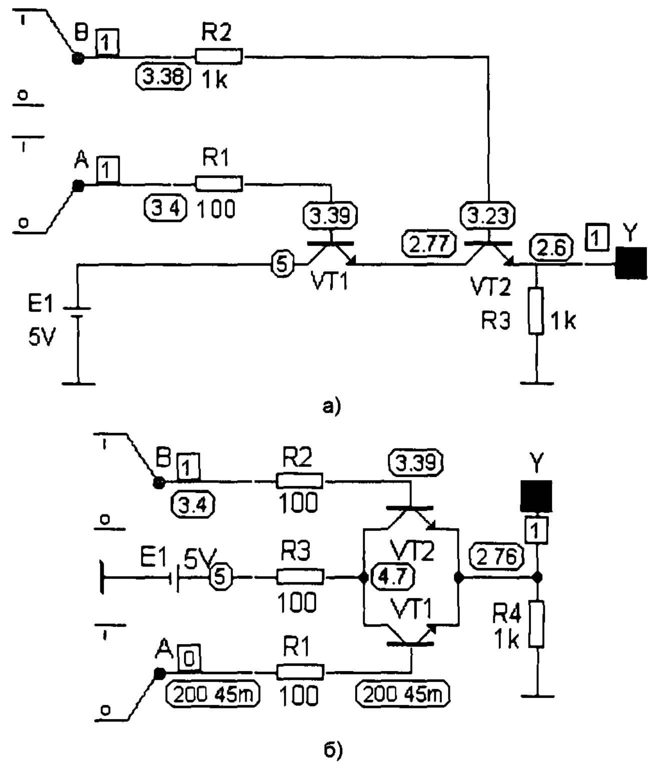


Рис 29 Резистивно-транзисторный ЛЭ: а – *И*; б – *ИЛИ* (**МС**)

зует логически полную систему. Соединяя последовательно элементы *ИЛИ* и *НЕ*, получим элемент *ИЛИ-НЕ*, по-английски Not OR, обозначаемый NOR. Этот элемент называют по фамилии американского ученого Чарльза Сандерса Пирса элементом Пирса ЛЭ *ИЛИ-НЕ* также образует полную систему. Этими свойствами ЛЭ пользуются при разработке микросхем

Резистивно-транзисторные ЛЭ Шеффера и Пирса в программе **EWB** можно получить из предыдущих схем рис. 27 и 28, если выходной сигнал снять не с эмиттера (повторение), а с коллектора (инверсия) В результате получатся схемы логических элементов, показанные на рис 30 и 31.

Там же показаны полученные на приведенных моделях соответствующие таблицы истинности. Сравнивая две таблицы на рис 30,а и 31,а с соответствующими таблицами на рис.11,б и

11,д, видим, что на тех же наборах переменных инвертируется только функция выхода Y (в отличие от перехода какого-либо элемента к отрицательной логике, сопровождающегося еще и инверсией всех аргументов, т е всей таблицы)

Описанные ЛЭ в программе **МС** показаны на рис 32

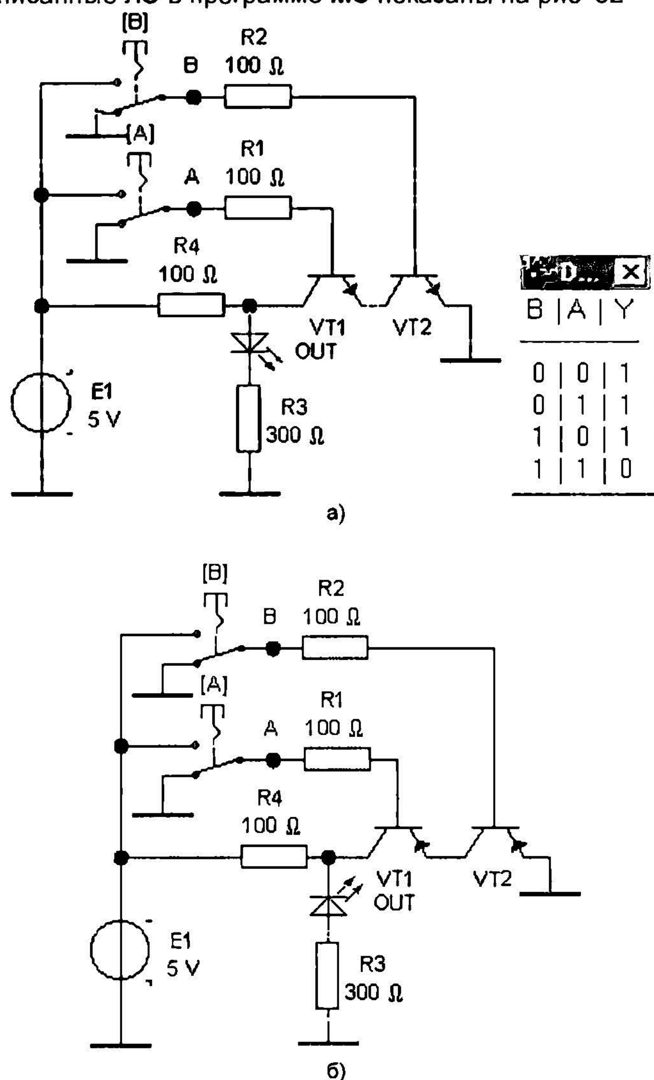
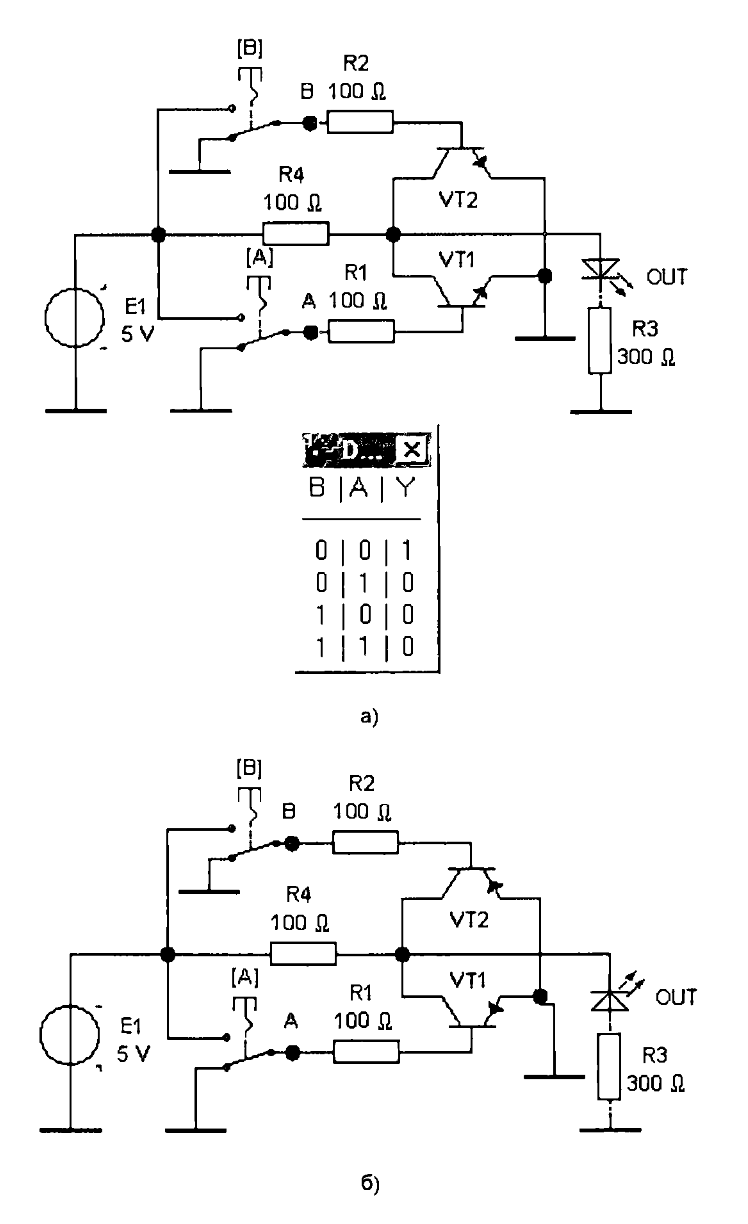


Рис. 30. Резистивно-транзисторный ЛЭ *И-НЕ*: а – при положительном; б – при отрицательном сигнале (**EWB**)



B	A	Y
0	0	1
0	1	0
1	0	0
1	1	0

Рис. 31 Резистивно-транзисторный ЛЭ *ИЛИ-НЕ* а – при положительном, б – при отрицательном сигнале (**EWB**)

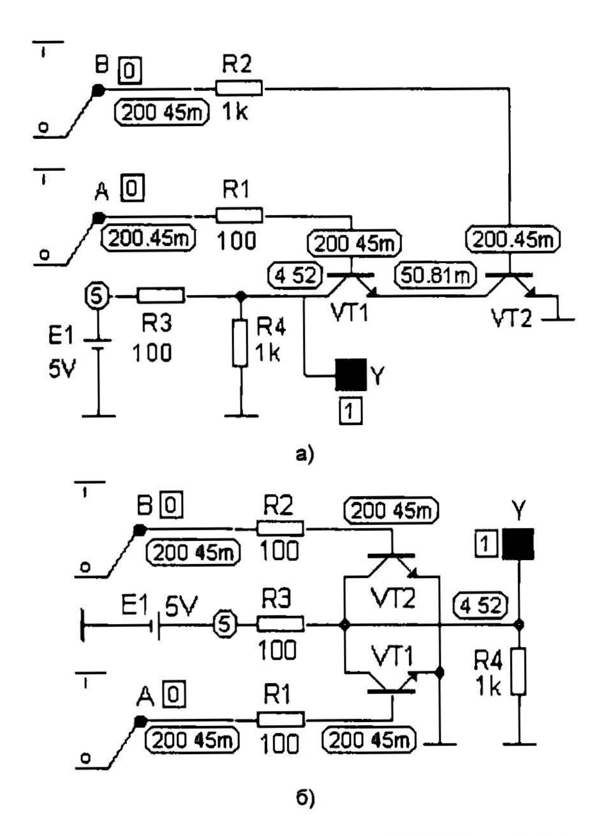


Рис 32 Резистивно-транзисторные ЛЭ а – *И-НЕ*, б – *ИЛИ-НЕ* (**МС**)

Диодно-транзисторная логика

Составные (комбинационные) ЛЭ можно выполнить, сочетая входной каскад на диодах с выходным на транзисторах. Это и будет так называемая диодно-транзисторная логика.

В качестве примеров диодно-транзисторной логики приведем схемы элементов Пирса и Шеффера Для этого воспользуемся сохраненными ранее файлами диодно-резистивной логики элементов *ИЛИ* и *И* (см рис 23) Скопируем перечисленные каскады на новый лист по командам. New>Schematic>Save As...>Open> Select All>Copy>Open>Past Далее схемы редактируются в соответствии с рис. 33. Разумеется, можно избрать иные варианты команд, выбрать другие номиналы или типы компонентов, нако-

нец, иные схемные решения. Что ж, как сказал поэт: «...твори, выдумывай, пробуй!».

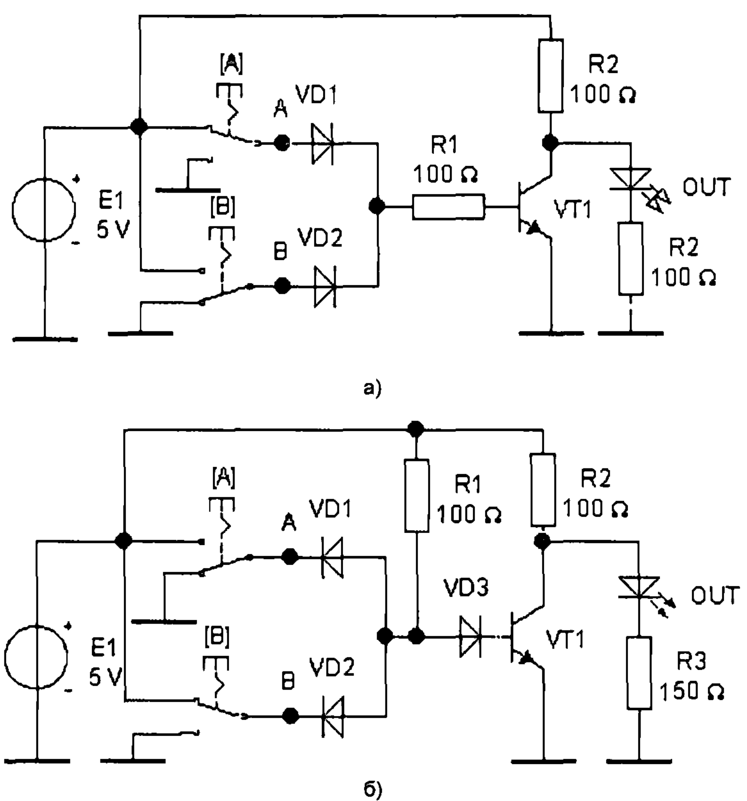


Рис 33 Диодно-транзисторные ЛЭ· а – *ИЛИ-НЕ*; б – *И-НЕ* (**EWB**)

Включим моделирование нажатием ЛКМ на I (In – включено) , и нажимаем на управляющие клавиши (при английской раскладке клавиатуры). Убеждаемся, что схема на рис. 33,а отрицает логическую операцию *ИЛИ*, если хотя бы на один вход подается лог 1 (+5 V), то СИД не горит: стрелки остаются не зачерненными.

Таким образом, это ЛЭ *ИЛИ-НЕ*. Схема на рис. 33,б отрицает логическую операцию *И*, если хотя бы на один вход подается лог 0 (0 В), то СИД горит (стрелки будут зачерненными). Таким обра-

зом, это ЛЭ *И-НЕ* Наличие дополнительного диода VD3 в этой схеме необходимо для вывода рабочей точки транзистора VT1 в режим насыщения В программе **МС** аналогичные ЛЭ *ИЛИ-НЕ* и *И-НЕ* диодно-транзисторной логики показаны на рис 34

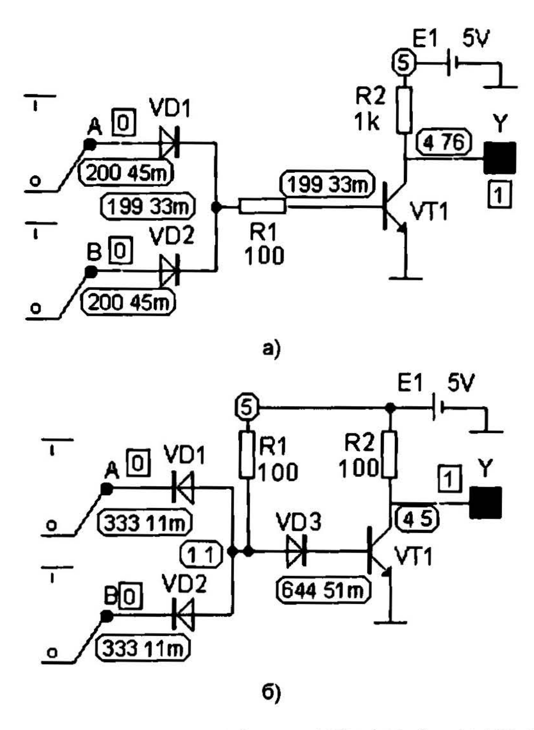


Рис 34 Диодно-транзисторные ЛЭ а – *ИЛИ-НЕ*, б – *И-НЕ* (**МС**)

Интегральные логические компоненты

Транзисторно-транзисторные микросхемы

ТТЛ микросхемы – один из наиболее распространенных типов микросхем (логических интегральных схем) Рассмотрим некоторые ЛЭ на уровне микросхемотехники

Во входном каскаде базового ТТЛ элемента *И-НЕ* используется многоэмиттерный транзистор (рис 35,а). В подобном NPN транзисторе по интегральной технологии выполняют от двух до восьми эмиттеров, что позволяет создать многовходовый ЛЭ с одновременным сокращением общего числа компонентов Сиг-

нал подается на эмиттеры, играющие роль входных диодов многоэмиттерного транзистора (рис 35,а) VD1 и VD2 в схеме ЛЭ *И* Коллекторный переход также можно представить диодом VD3, играющим роль смещающего в той же схеме В результате приходим к диодной модели многоэмиттерного транзистора (рис 35,б)

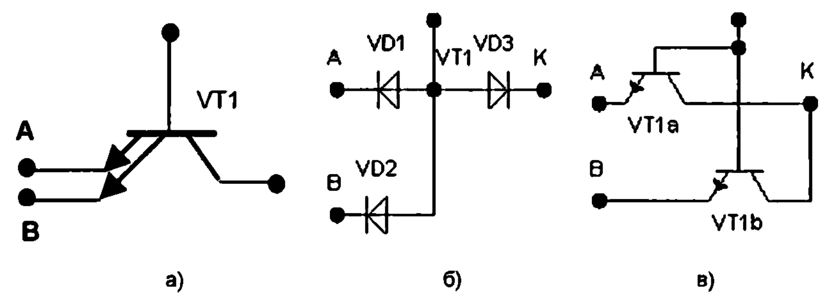


Рис 35 Многоэмиттерный транзистор
а – УГО, б – диодная модель, в – транзисторная модель (**EWB**)

Модель двухвходового многоэмиттерного транзистора можно также выполнить в виде составного транзистора из двух транзисторов VT1a и VT1b (рис 35,в) Эти упрощенные модели не отражают многие процессы в микросхеме, но позволяют в принципе рассмотреть схемотехнические решения ЛЭ Используя указанные модели и инвертор на одном транзисторе, приходим к упрощенным моделям ЛЭ *И-НЕ* типа ТТЛ (рис 36)

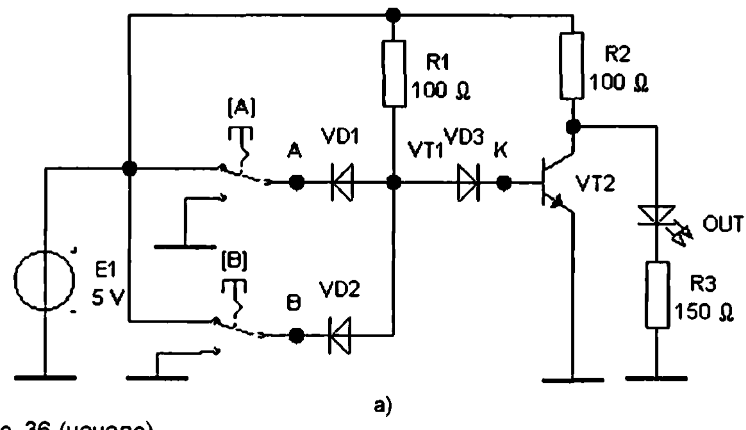


Рис 36 (начало)

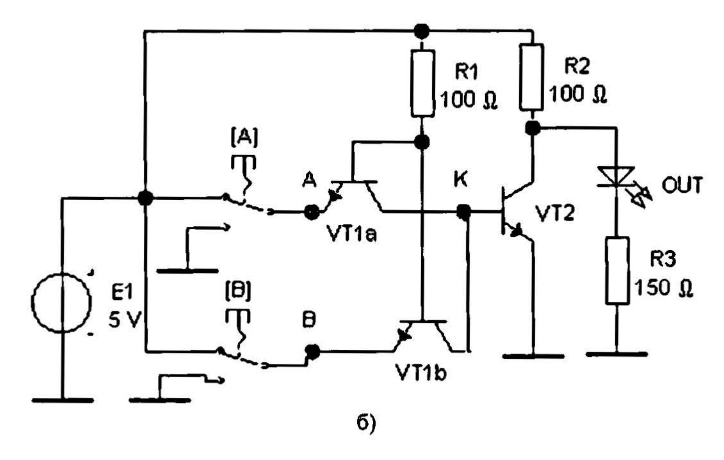


Рис 36 (окончание) Упрощенные модели элемента *И-НЕ* типа ТТЛ (**EWB**)

Реальный ЛЭ имеет более сложную структуру и состоит из трех каскадов входной на многоэмиттерном транзисторе VT1 (VT1a и VT1b), реализующий функцию *И*, фазоразделительный на транзисторе VT2 с возможностью реализации на нем функции *ИЛИ* и выходной усилитель на двух транзисторах VT3 и VT4 (рис 37)

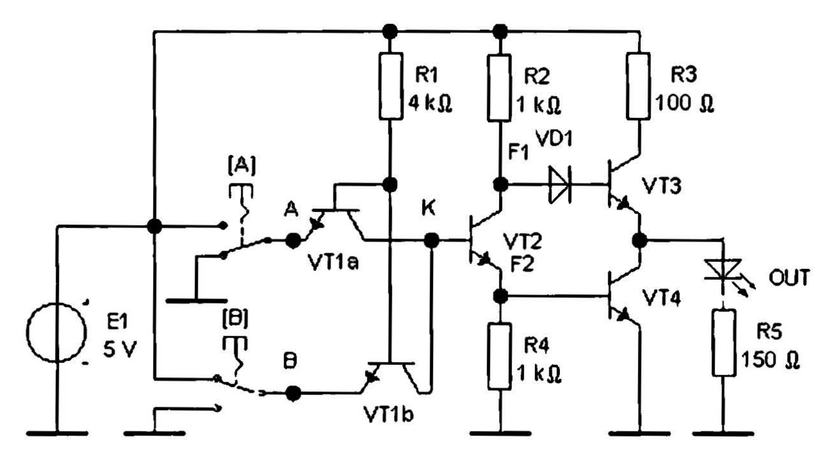
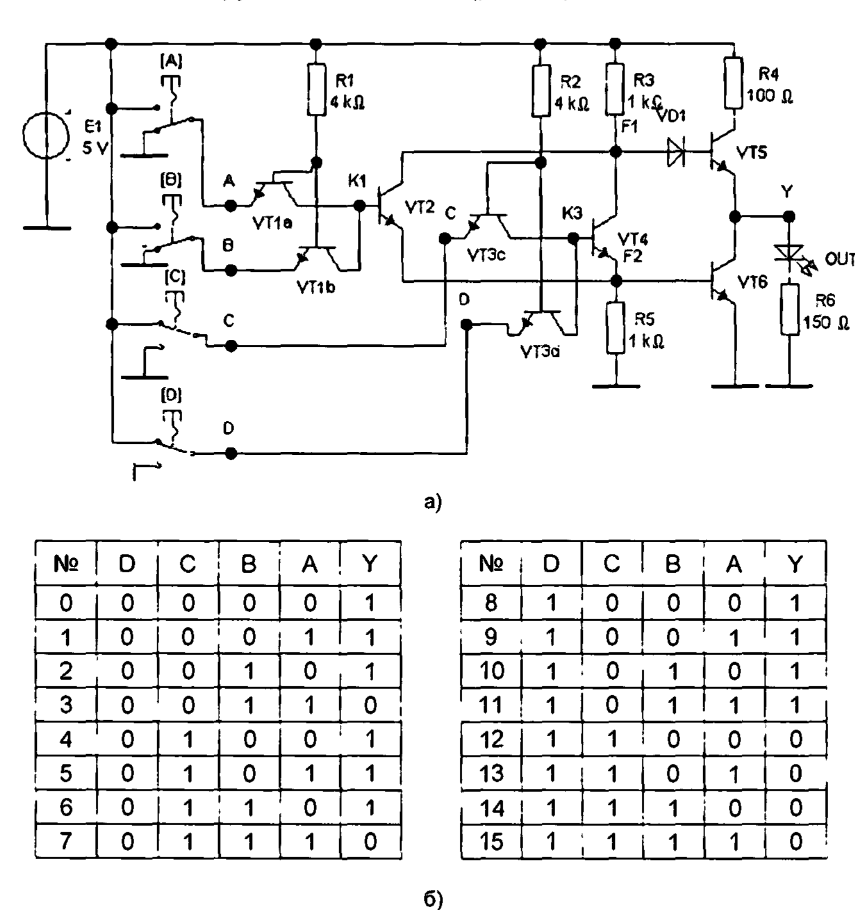


Рис 37 Модель базового элемента *И-НЕ* микросхемы типа ТТЛ (**EWB**)

Данный базовый элемент (по англоязычной терминологии ТТЛ-вентиль) является «функциональным кирпичиком», используемым при построении большинства логических схем ряда серий микросхем ТТЛ Наличие фазоразделительного каскада на транзисторе VT2 позволяет наращивать входные каскады, выполняя расширение по схеме *ИЛИ* Добавив в предыдущей схеме еще один входной каскад, получим модель ЛЭ, выполняющего функцию *И-ИЛИ-НЕ* (рис 38)



№	D	C	B	A	Y
0	0	0	0	0	1
1	0	0	0	1	1
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	0

№	D	C	B	A	Y
8	1	0	0	0	1
9	1	0	0	1	1
10	1	0	1	0	1
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

б)

Рис 38 Элемент *И-ИЛИ-НЕ* микросхемы типа ТТЛ
а – схема модели, б – таблица истинности (**EWB**)

Совет При сборке схем не пытайтесь воспроизводить плотный монтаж, показанный на рисунках: реально эти схемы получены значительно при менее плотной сборке, а уже затем уплотнены графическим редактированием.

Теперь выходной сигнал будет зависеть от четырех переменных A, B, C, D. Следовательно, ЛФ будет иметь четыре аргумента, т е. Y=(A,B,C,D). Для исследования модели заготовим ТИ, она должна содержать все наборы этих четырех переменных, каждая из которых принимает значения 0 и 1 всего 2^4=16 состояний Счет начинается с нуля. Если по-прежнему придерживаться упорядоченного выбора наборов аргументов как двоичных слов, то ТИ примет вид, показанный на рис 38,б Для удобства работы ТИ набрана в программе Excel Поясним ее структуру Левый боковик под № представляет номера состояний и одновременно десятичные числа от 0 до 15 (всего 16 чисел). Далее в головке ТИ следуют разряды двоичного числа самый старший D(8),затем C(4), B(2) и, наконец, самый младший A(1) Нетрудно заметить, что при такой записи в колонке A сверху вниз чередуются 0 и 1, в колонке B 0,0 и 1,1, и т д. Поэтому в программе Excel можно быстро заполнить эту часть ТИ, используя сервисную функцию копирования групп ячеек. Десятичным числам № в строке справа от них под DCBA соответствует двоичное число Для того чтобы заполнить колонку Y, необходимо после сборки схемы включить моделирование и «честно» перебирая ключами 16 вариантов фиксировать соответствующие показания СИД в виде не горит – 0, горит – 1. Целесообразно также произвести проверку по формуле ЛФ Для ЛЭ *И-ИЛИ-НЕ* при четырех переменных она имеет вид.

Y=(AB + CD)`

Необходимо опять-таки «честно» подставлять соответствующие нули и единицы и, перебирая все 16 вариантов, подсчитывать в уме результат При этом надо помнить, что в булевой алгебре, как и в обычной, 0*1=1*0=0, 0*0=0, 1*1=1, но **1+1=1** и **0`= 1**, а **1`=0** (здесь, как и ранее, значком штрих ` отмечена инверсия).

С увеличением числа компонентов в отдельной микросхеме или с увеличением числа ЛЭ в схеме, выполняющей сложную функцию, быстродействие сильно уменьшается. На физическом уровне время переключения из одного состояния в другое определяется временем рассасывания носителей заряда, которое может быть сильно уменьшено с использованием так называемых диодов Шоттки

Вальтер Шоттки (Schottky) – немецкий физик, родившийся в Швейцарии, выполнил многочисленные исследования в области физики твердого тела, термодинамике и электронике Ему принадлежит изобретение экранной сетки в электровакуумных лампах, открытие барьера Шоттки, в 1929 г он ввел в научно-технический обиход понятие о «дырках» в полупроводниках Примерно до 80-х годов в русскоязычной литературе фамилию Schottky писали с двумя буквами «т», следуя немецкому оригиналу· Шоттки Кому взбрело в голову экономить на одной букве непонятно, но сейчас некоторые авторы и даже энциклопедии фамилию Schottky на русском языке пишут с одним «т» Нам же представляется, что более правильна прямая побуквенная транслитерация с двумя «тт» Обращаясь к ономастике (от греч onoma – имя, название) – науке, изучающей имена собственные, нетрудно обнаружить в фамилии Schottky лексическое значение, связанное с ее происхождением В немецком языке Schotte – шотландец Если убрать одну букву «t», то в обратном переводе придется считать, что его фамилия происходит от слова Schote, означающего стручок! К такой конвенции мы присоединиться никак не можем: нельзя же быть «иванами родства не помнящими».

Обычно контакт металл – полупроводник является не выпрямляющим (поэтому его часто называют омическим). Однако, как установил Шоттки, в обедненном основными носителями приконтактном слое полупроводника возникает энергетический барьер, который может играть роль выпрямляющего устройства. В диоде, с барьером Шоттки, в отличие от диодов с P-N переходом, ток обусловлен только основными носителями Отсюда для диодов с малой площадью время переключения имеет порядок сотых долей наносекунды, и соответственно рабочие частоты составляют десятки гигагерц.

В УГО диода Шоттки катод изображают в виде вытянутой латинской буквы S – заглавной буквы фамилии ученого.

В микросхемах используют транзисторы Шоттки, представляющие собой как бы обычный транзистор, между базой и коллектором которого включен диод Шоттки. В интегральной технологии этот «гибрид» изящно получается простым продолжением напыляемой полоски алюминия. все гениальное просто! Вообще же способ подобного включения диодов как элементов нелинейной обратной связи, устраняющей насыщение транзисторов, был предложен Б. Н. Кононовым еще в 1955 г.

Условную схему транзистора Шоттки покажем в программе **MC**. Задав команды: Component>Analog Library>Diode>Schottky

>1N571, выведем на рабочее поле диод VD1, и далее достроим схему согласно приведенному рисунку (рис. 39).

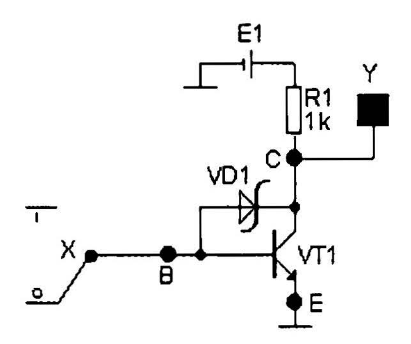


Рис. 39 Элемент *НЕ* с диодом Шоттки (**МС**)

В интегральном транзисторе Шоттки выводы С, В, Е имеют смысл коллектора, базы и эмиттера соответственно. Дополнительно в число символов транзисторно-транзисторных логических микросхем, выполненных на подобной элементной базе, добавляют букву Ш: ТТЛШ (на английском – S-TTL).

В англоязычной и соответственно в переводной литературе базу транзисторов Шоттки, так же как и катод в диодах Шоттки, в развернутых схемах микросхем изображают в виде знака, напоминающего знак математического интеграла: $\int$. Этот символ, как известно, также представляет вытянутую букву S, так как происходит от английского Square – площадь (ведь геометрический смысл интеграла собственно и заключается в нахождении площади под кривой). В данном же случае получился своеобразный графический каламбур: интегралом изображается элемент, получаемый по интегральной технологии. К сожалению, в стилистике УГО отечественных ГОСТов вся эта лирика исчезает, так как символ выполняют с прямоугольными окончаниями, по очертаниям которых вспомнить Schottky или представить себе интеграл весьма затруднительно.

Диоды Шоттки используют в различных микросхемах. В качестве примера на рис. 40 показана схема ТТЛШ, выполняющая функцию *И-НЕ* Для индикации логического состояния выхода Y в данной схеме использован специальный чувствительный индикатор типа IO_STD_ST, выбираемый из библиотеки компонентов.

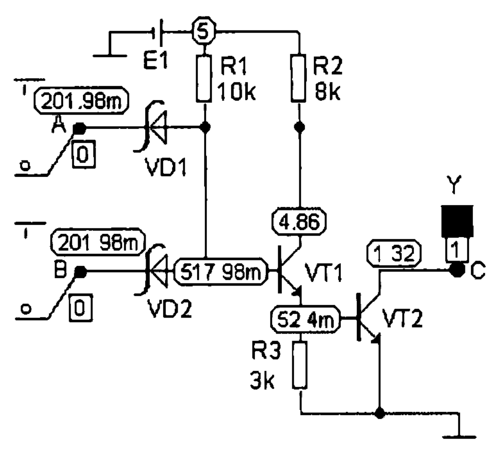


Рис. 40 ЛЭ ТТЛШ *И-НЕ* с открытым коллектором (**МС**)

Особенностью этой схемы является то, что транзистор VT2 имеет «открытый коллектор» – точка С. Это позволяет объединять между собой ЛЭ по принципу «монтажное *ИЛИ*». Возникновение этого термина связано с тем, что логическая функция образуется не в одном ЛЭ, а при соединении (монтаже) группы ЛЭ Вначале подобные схемы использовались для включения различных исполнительных элементов через реле и многочисленные системы индикаторных лампочек, отслеживающих работу компьютера. Безусловно, это было захватывающее зрелище, напоминающее праздничную иллюминацию, но позже их стали применять для более важной цели: подключения к «общей шине» десятков различных устройств.

При использовании системной магистрали компьютера несколькими передатчиками и приемниками сигнала возникает необходимость их перевода в неактивное состояние. Это состояние микросхем называют «высокоимпедансным» (от англ. impedance – полное или кажущееся сопротивление цепи на переменном токе) Однако по ГОСТу этот термин был не рекомендован к употреблению. Используют также термин ЛЭ с «тремя состояниями» (на англ. TRI-STATE, что является одновременно товарным знаком американской фирмы National Semiconductors Corp., создавшей их). Это тоже не очень удачно, так как логических состояний конечно два, а данный случай является другим режимом работы. За неимением лучших, придется пользоваться этими обиходными

терминами Итак, на выходе микросхемы, либо 0, либо 1, либо этот выход выключен В последнем случае режим выхода – «обрыв», при этом сопротивление между выходом и общей шиной (сигнальная земля), а также и между выходом и шиной питания достигает единиц МОм Для иллюстрации работы подобных устройств соберем условную схему ЛЭ *НЕ* по входу X и добавим в нее вход Z, разрешающий или, соответственно, запрещающий выход сигнала Y (см рис 41).

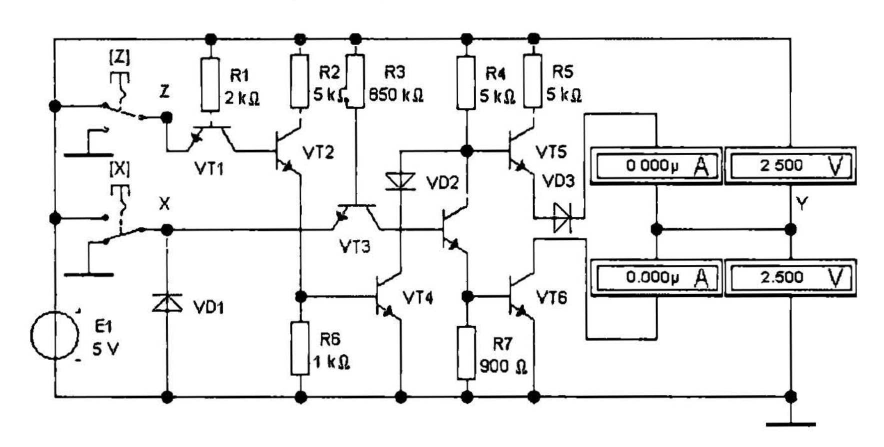


Рис. 41. ТТЛ «TRI-STATE» (**EWB**)

Для выявления высокоимпедансного режима, в разрыв выхода включены два амперметра, а между выходом и соответствующими шинами вольтметры Эти приборы в программе **EWB** извлекаем из панели Indicators. Как видно из приведенного рисунка, при Z=1 для любых X, ток равен нулю, а напряжения составляют половину E1, что достигается переводом в режим отсечки и ключевого, и выходных транзисторов.

Эмиттерно-связанная логика

Цифровые микросхемы эмиттерно-связанной транзисторной логики представляют собой переключательные схемы с объединенными эмиттерами. Междукаскадные связи в микросхемах эмиттерно-связанной логики являются непосредственными. Подобные схемы также обладают большим быстродействием, так как транзисторы в них работают в ненасыщенном (линейном) режиме

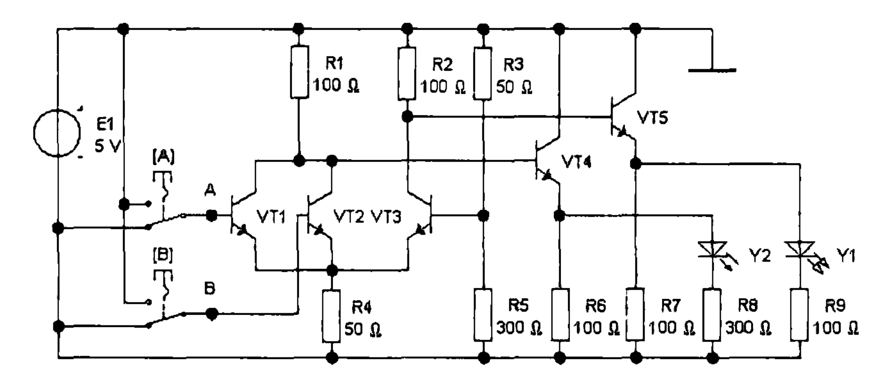


Рис 42 Элемент *ИЛИ/ИЛИ-НЕ* микросхемы эмиттерно-связанной логики (**EWB**)

Соберем схему ЛЭ в программе **EWB** (рис. 42) В ней входные транзисторы VT1 и VT2 и транзистор VT3 образуют дифференциальный каскад, R3 и R5 играют роль встроенного источника в режиме эмиттерных повторителей Таким образом, схема имеет высокое входное и низкое выходное сопротивление. Положительный полюс источника питания E1 заземлен Данный ЛЭ выполняет логическую функцию *ИЛИ* на выходе Y1 и *ИЛИ-НЕ* на выходе Y2.

Аналогичную схему можно собрать и в программе **МС**, введя небольшие и не принципиальные изменения, связанные со спецификой отсчета уровней и их индикацией (рис. 43).

Реальные устройства отличаются от этих моделей по параметрам компонентов, кроме того, есть и схемотехнические различия. Так, например, они имеют большее число входов (4 или 8). Источник опорного напряжения обычно выполняют по схеме параметрического стабилизатора. Выходные каскады подключают к специальным источникам смещения уровня, обеспечивая малое выходное сопротивление и согласование выходных и входных уровней ЛЭ при их совместной работе и возможность непосредственно подавать с них сигналы на кабель с волновым сопротивлением 50 Ом. Для улучшения рабочих характеристик используют сложные древовидные структуры, выполненные по интегральной технологии (см. например, библиотечный файл Eclgate.CIR в программе **МС**)

Всякое моделирование условно и здесь, если специально не оговаривается иное, главное внимание уделяется логическому поведению цифровых устройств.

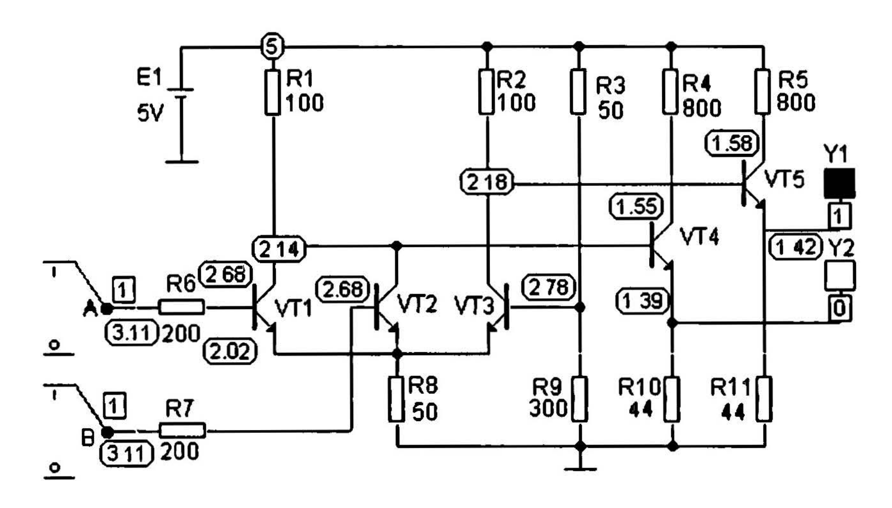


Рис. 43. Элемент *ИЛИ / ИЛИ-НЕ* микросхемы эмиттерно-связанной логики (**МС**)

Интегральная инжекционная логика

Логические микросхемы с инжекционным питанием являются дальнейшей модификацией транзисторной логики с непосредственными связями. Средствами интегральной технологии из них не только как бы удалено все лишнее, ухудшающее рабочие характеристики, но и произведено сращивание элементов отдельных биполярных транзисторов. В кристалле так называемые горизонтальные транзисторы VT1 (*p-n-p* типа) и вертикальные VT2 (*n-p-n* типа) совмещаются при изготовлении так, что область базы n1 первого транзистора VT1 является одновременно эмиттером транзистора VT2, область базы которого p2 в свою очередь служит коллектором для VT1 (рис. 44,а).

Транзистор VT1, включенный по схеме с общей базой, играет роль индивидуального генератора тока каскада и работает в режиме двойной инжекции (насыщения), так как его коллектор имеет потенциал больше нуля.

Эмиттерная область p1, подключенная к положительному полюсу источника питания E1, вводит в эту область неосновные неравновесные носители и называется инжектором. Слово «инжектор» пришло к нам из французского языка: «injecteur», где было в свою очередь заимствовано из латинского – «injicio», озна-

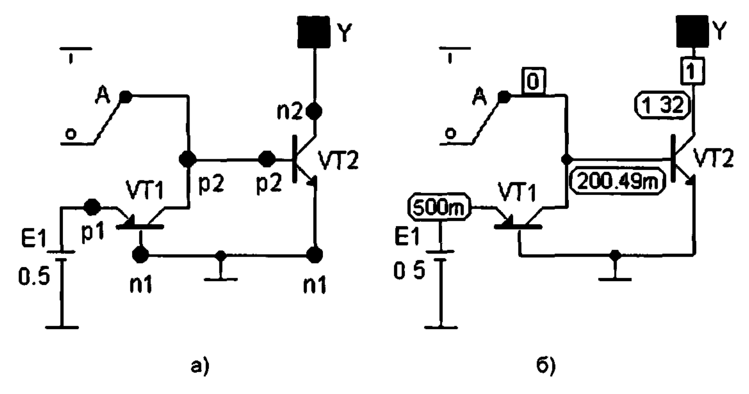


Рис 44 Схема транзистора с инжекционным питанием (МС)

чающего «вбрасываю». (Сейчас это слово активно входит в наш обиход в связи с широким распространением автомобильных инжекторных двигателей. В последних, разумеется, впрыскиваются не носители заряда, а бензин).

Наличие такого специфического источника питания и определило название этого класса микросхем: Интегральная Инжекционная Логика – И^{2}Л (на английском Injection-Logic Circuitry – I^{2}L) В реальных ЛЭ И^{2}Л вертикальный *n-p-n* транзистор имеет несколько коллекторов, являющихся логическими выводами элемента.

Работа транзистора с инжекционным питанием продемонстрирована на примере эквивалентной схемы в программе **МС** на рис. 44,б. Здесь показан режим инверсии сигнала: A=0, Y=1. В схеме использован источник питания с пониженным напряжением (E1=0.5 V) и в связи с этим специальный чувствительный индикатор Y (типа IO_STD_ST), выбираемый из библиотеки компонентов.

Наличие многоколлекторного выхода в транзисторах позволяет выполнять на их основе различные ЛЭ. На рис 45 показана схема с двумя каскадами, соединенными выводами коллекторов в точке С Такое соединение в схемотехнике носит название «монтажное *И*»

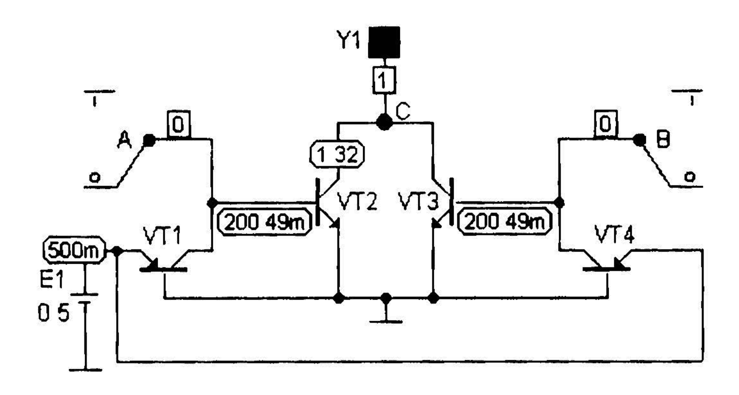


Рис 45 Логический элемент *ИЛИ - НЕ* инжекционного типа (**МС**)

Действительно, если в схемах транзисторно-транзисторной и диодно-транзисторной логики операция *И* выполняется с помощью многоэмиттерного транзистора или диодной сборки, то здесь она выполняется путем соединения инверторов металлическими проводниками. В результате на выходе образуется сигнал Y=A`B`. В этом можно убедиться, поэкспериментировав со схемой, показанной на рис. 45

Используя диоды Шоттки на выходе ЛЭ инжекционного типа, можно ограничить амплитуду выходного сигнала на заданном уровне (рис 46)

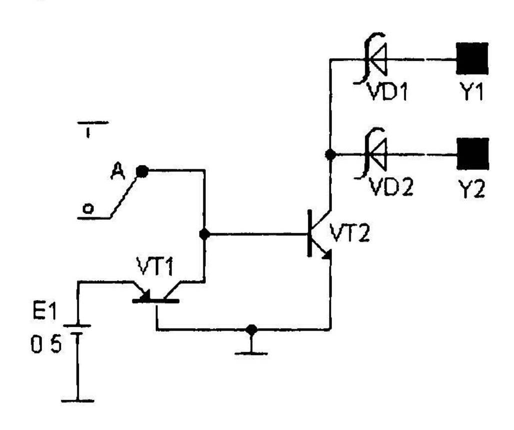


Рис 46 Схема фиксации с диодами Шоттки (**МС**)

Отсутствие резистивных элементов и совмещение областей биполярных транзисторов в ЛЭ И^{2}Л позволило существенно увеличить плотность компоновки микросхем, уменьшить время переключения и энергопотребление Поэтому подобные микросхемы нашли широкое применение как в разнообразной бытовой, так и в промышленной электронике. Это и электронные часы, и цифровые блоки настройки радио- и телевизионных приемников, а также многочисленные цифровые устройства (ЦУ) компьютерной техники Особенно важно, что подобные схемы обладают высокой помехоустойчивостью

Логические элементы на МОП-транзисторах

Широкое распространение в цифровых устройствах получили микросхемы на полевых транзисторах В этих транзисторах в некоторой области полупроводника (канале Р-типа или N-типа) протекает ток основных носителей заряда одного знака При изготовлении транзисторов возможны два типа легирования канала обеднение и обогащение Управление величиной тока осуществляется поперечным электрическим полем отсюда название – полевые транзисторы (на англ. – Field Effect Transistor, сокращенно FET).

Полевые транзисторы с изолированным затвором – Insulated (изолированный) Gate (затвор), т е Insulated Gate FET или сокращенно IJFET, содержат в своей структуре металл-оксид-полупроводник Отсюда происходит сокращенное название МОП или, на английском, Metall-Oxide-Semiconductor FET (MOSFET).

Схемотехнические решения ЛЭ на МОП-транзисторах существенно не отличаются от схем на биполярных транзисторах В программе **МС** выполним команды: Component>Analog Primitives> Analog Devices>NMOS В результате на экране появится УГО полевого транзистора MOS с каналом N-типа (рис. 47,а).

В символике условных графических обозначений полевых транзисторов электроды имеют следующие названия: затвор (Gate) – аналог базы, исток (Source) – аналог эмиттера, сток (Drain) – аналог коллектора, кроме того, имеется вывод от подложки кристалла (Body). Собрав на полевых транзисторах схемы инвертора (рис. 47,б) и буфера (рис. 47,в), убеждаемся, что они функционируют аналогично соответствующим устройствам на биполярных транзисторах (см. рис. 24 и 26).

В программе **EWB**, нажав на пиктограмму с изображением транзистора (Transistors) , выбираем полевой транзистор

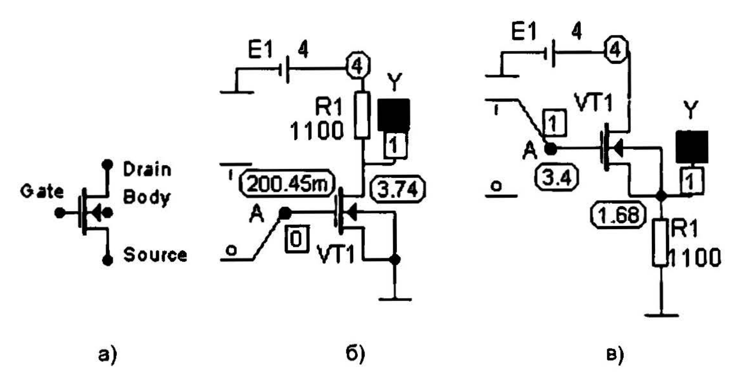


Рис 47 ЛЭ на МОП-транзисторе
а – УГО NMOS-транзистора; б – инвертор; в – буфер (**МС**)

типа 4-Terminal Depletion N-MOSFET (четырехвыводной с обеденным N-каналом). В этой программе УГО несколько отличается изображением затвора. Для индикации логических уровней выхода теперь используем логический пробник (тот же СИД или LED). Нажав на пиктограмму Indicators (индикаторы), выбираем компонент, называемый Red Probe (дословно – красный пробник). Этот пробник имеет несколько своеобразное изображение, напоминающее надувной шарик на веревочке, который ослик Иа-Иа подарил Винни-Пуху. «От великого до смешного один шаг», но на самом деле это весьма полезный виртуальный индикатор, которым мы далее будем широко пользоваться. Индикатор имеет один вывод, которым и включается в контролируемую точку схемы (так же, как соответствующий LED в виде транспаранта в программе **МС**). Цвет индикатора по умолчанию принят красным, если уровень напряжения высокий и белым – при низком. Цвет индикации можно сделать зеленым или синим, войдя в меню его свойств. Так что из подобных индикаторов при желании можно создать на экране красочное динамичное панно.

Соответствующие схемы инвертора и буфера в программе **EWB** показаны на рис. 48

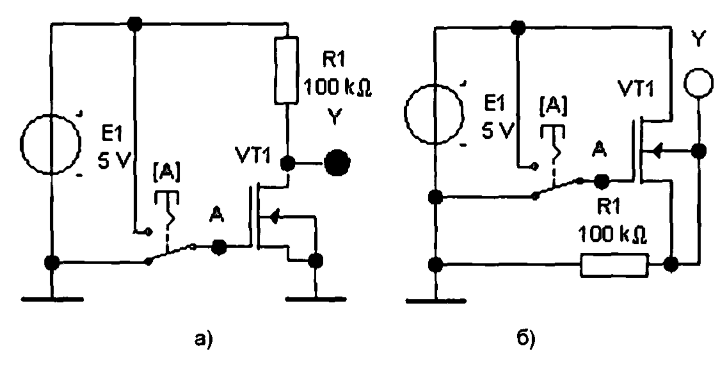


Рис 48 ЛЭ на МОП-транзисторе а – инвертор, б – буфер (**EWB**)

Далее, соединяя полевые транзисторы последовательно и параллельно, получаем схемы ЛЭ *И* и *ИЛИ*, показанные на рис 49 и 50. В программе **МС** требуется специальный выбор типа транзистора и индикатора из соответствующих библиотек, открываемых при редактировании свойств этих компонентов. Выбранные типы указаны на схеме (см рис. 50)

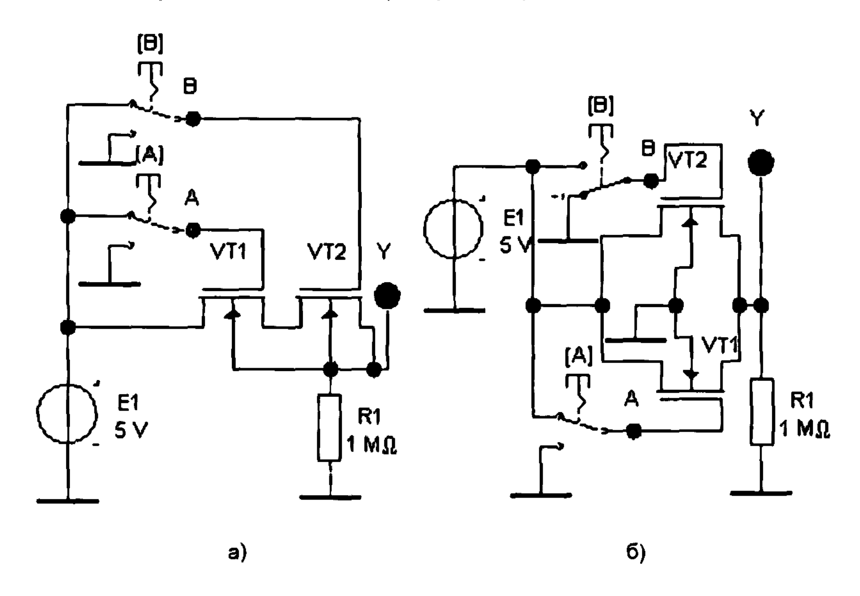


Рис 49 ЛЭ на МОП-транзисторах: а – *И*; б – *ИЛИ* (**EWB**)

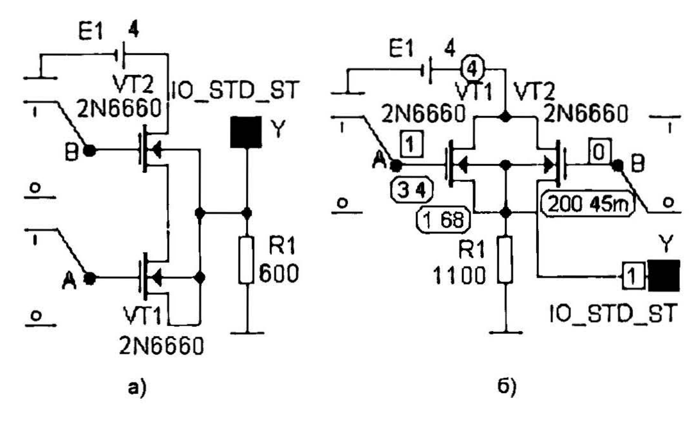


Рис 50 ЛЭ на МОП-транзисторах а – *И*, б – *ИЛИ* (**МС**)

Приведенные выше схемы (рис 47–50) содержали нагрузочные резисторы. В микросхемах на МОП-транзисторах резисторы не используются, так как привели бы к ухудшению всех характеристик Место этого нагрузочного резистора занимает соответствующий полевой транзистор, получаемый по той же интегральной технологии Сопоставляя схемы инверторов на рис 47,а и 48,а с аналогичными на рис 51, видим, что на них вместо резистора R1 включен транзистор VT2

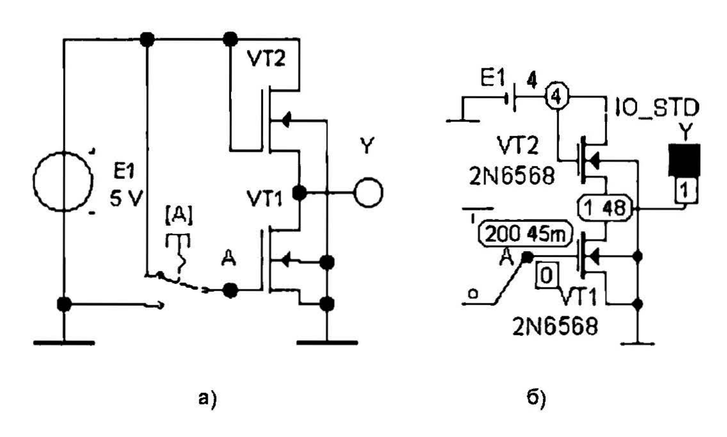


Рис 51 ЛЭ *НЕ* на МОП-транзисторах а – **EWB**; б – **МС**

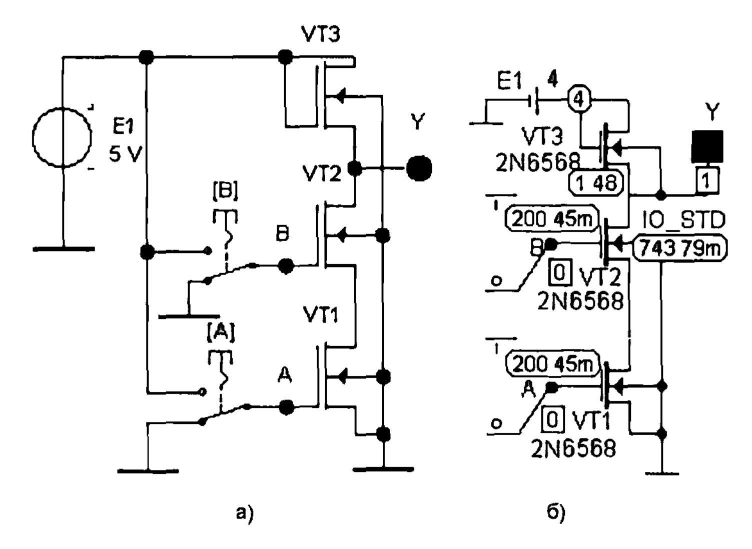


Рис 52 ЛЭ *И-НЕ* на МОП-транзисторах а – **EWB**; б – **MC**

Точно так же выполняем схемы ЛЭ *И-НЕ* и *ИЛИ-НЕ* и проводим моделирование их работы (см. рис. 52 и 53) Видим, что все эти схемы по существу представляют те же комбинации простейших ключей, с которых начиналось рассмотрение работы логических цепей

Совет После сборки схем для проведения моделирования их работы в программе **EWB** надо нажать на клавишу I/0, а в программе **MC** войти в режим Transient>Run>Node Numbers Не забывайте выходить из этих режимов после получения результатов. Если же схемы не работают должным образом, то проверьте правильность «монтажа», типы использованных компонентов и их номиналы, а также режимы анализа. Наконец, загляните в опцию «Help». О некоторых ошибках могут поступать сообщения от сервисной части программ.

После графического редактирования схем полезно произвести очистку экрана от различного «мусора»: удалить все лишние компоненты и выполнить команды: Window>Arrange или с клавиатуры Ctrl+W.

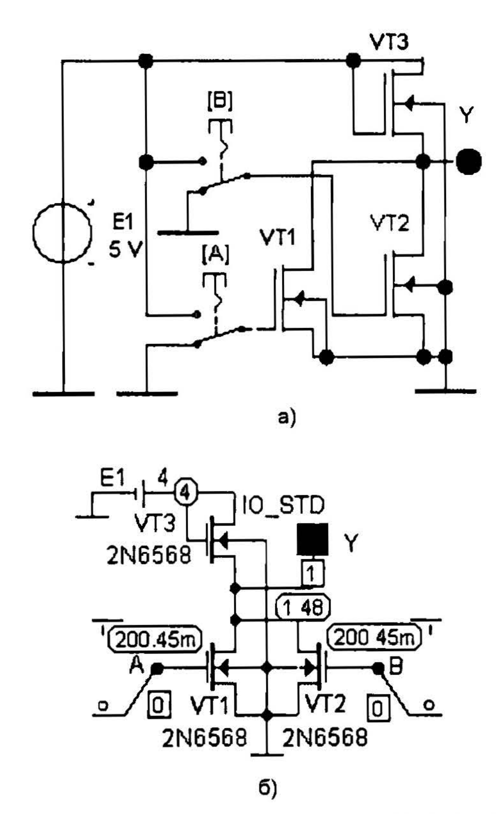


Рис 53. ЛЭ *ИЛИ-НЕ* на МОП-транзисторах. а – **EWB**, б – **MC**

Логические элементы на КМОП-транзисторах

Использование комплементарных структур добавляет в русской аббревиатуре их названия префикс «К»: КМОП или в английском «С» (от Complementary): CMOS. Именно последний акроним используется для обозначения энергонезависящей памяти компьютера, выполненной в виде интегральной микросхемы по соответствующей технологии Прилагательное «комплЕментарный» происходит от английского слова complement, которое в свою очередь происходит от латинского complementum, означающего пополнение, дополнение. (Не путайте это слово с его паронимом, близким по звучанию (омофоном)˙ «комплИмент», которое происходит от французского compliment и имеет совсем иной смысл)

Совмещая два комплементарных (дополнительных) друг другу МОП-транзистора, например, с каналами N-типа и P-типа, можно составить КМОП-инвертор.

В программе **EWB** выбираем, как и прежде, полевой транзистор типа 4-Terminal Depletion N-MOSFET (четырехвыводной с обеденным N-каналом) и, буквально рядом с ним, комплементарный ему полевой транзистор типа 4-Terminal Depletion P-MOSFET (четырехвыводной с обеденным P-каналом). УГО последнего отличается направлением стрелки в затворе (см транзистор VT2 на рис 54,а)

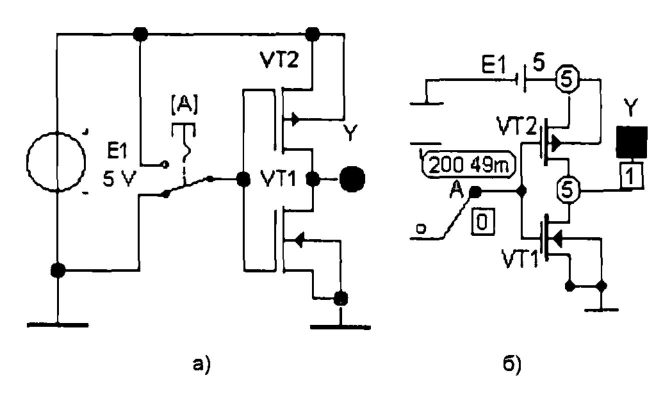


Рис 54 ЛЭ *НЕ* на КМОП-транзисторах: а – **EWB**; б – **MC**

Поскольку эти транзисторы двойственны (зеркальны) по типу канала, их питание также зеркально. Для придания нужной ориентации (у P-транзистора исток подключается к «плюсу») транзистор поворачиваем вокруг горизонтальной оси после его выделения с помощью кнопки Flip Vertical. Заменяя в схеме по рис. 51,а VT2 на комплементарный, получаем КМОП-инвертор (см. рис. 54,а)

В программе **MC** P-канальный транзистор выбираем по командам: Component>Analog Primitives>Analog Devices>PMOS. Далее, аналогично предыдущему, собираем схему КМОП-инвертора (см. рис. 54,б). Задавая в этих схемах на входном переключателе 0, на выходе получаем 1, и наоборот.

Совершенно аналогично собираем КМОП-схемы ЛЭ *И-НЕ*, а также *ИЛИ-НЕ* (см рис 55 и 56) и, перебирая варианты входных сигналов ключами A и B, сверяем их работу с соответствующими таблицами истинности (см рис 30,а и 30,б)

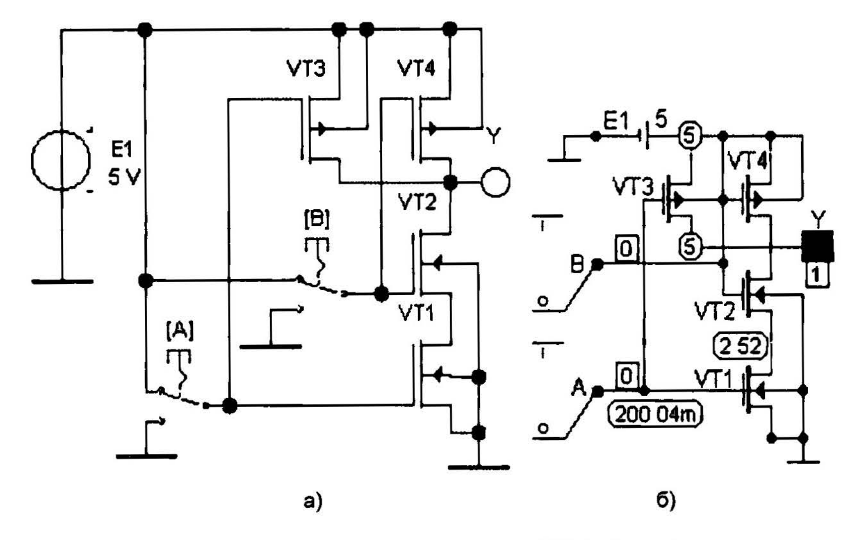


Рис 55 ЛЭ *И-НЕ* на КМОП-транзисторах: а – **EWB**, б – **МС**

Рассматривая геометрию (а правильнее, ввиду отсутствия метрики, топологию, т е «безразмерную геометрию») электронных схем, можно обратить внимание на то, что эти схемы могут преобразовываться друг в друга в соответствии с ранее описанными правилами Например, схема на КМОП-транзисторах положительной логики *ИЛИ-НЕ*, в которой N-канальные транзисторы включены параллельно, а Р-канальные последовательно (см рис. 56), преобразуется в схему на КМОП-транзисторах отрицательной логики *ИЛИ-НЕ*, если Р-канальные транзисторы включить параллельно, а N-канальные последовательно.

Используя подобное ярусное соединение КМОП-транзисторов, получают многовходовые микросхемы с очень малым энергопотреблением. Обладая рассмотренными выше простейшими базовыми элементами, можно создать схемы, выполняющие более сложные заданные логические функции и реализовать их в той или иной интегральной структуре.

КМОП-технология интегральных микросхем позволяет разрабатывать многофункциональные ЛЭ, отличающиеся высокой гибкостью в выборе вариантов построения компонентов ЦУ, получающихся на одних и тех же структурах.

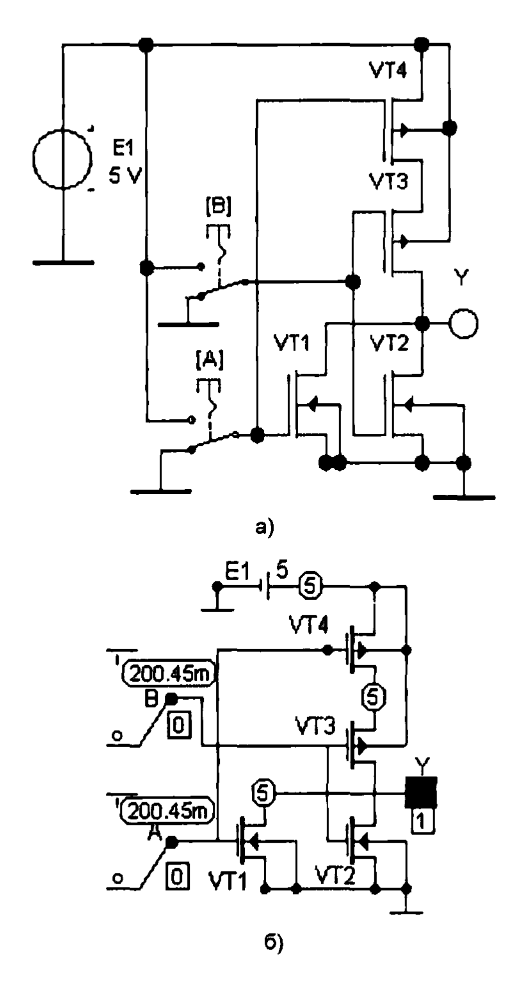


Рис. 56. ЛЭ *ИЛИ-НЕ* на КМОП-транзисторах: а – **EWB**; б – **MC**

Совет Прежде чем модернизировать предложенные схемы или придумывать новые, добейтесь, чтобы работали аналогичные типовые Цените свой труд: сохраняйте файлы с основными отработанными схемами в отдельной папке, систематизируя их на свой вкус. Они могут быть многократно использованы, в том числе и как части других схем.

Принципиально существуют два различных класса цифровых систем комбинационные, не зависящие от времени и предшествующих состояний, и последовательностные (или последовательные), обладающие памятью Логика работы соответствующих систем показывается далее с помощью специальных виртуальных инструментов компьютерных программ **EWB** и **MC**

1.3. Логический конструктор цифровых схем

Отыскивания законов физики – это вроде датской игры в кубики из которых нужно собрать целую картинку

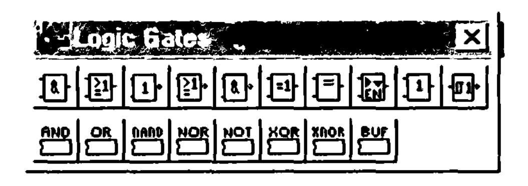
Р. Фейнмен

В обеих программах имеются специальные наборы базовых логических элементов

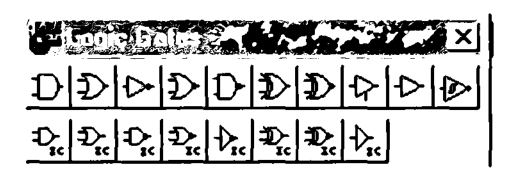
В программе **EWB** базовые логические элементы (БЛЭ) выбираются из подменю, открывающегося при нажатии на пиктограмму Logic Gates (логические элементы) со значком & . (так называемый ampersand) Этот символ был введен в письменность секретарем знаменитого древнеримского оратора Цицерона Марком Туллием Тироном Последний является изобретателем скорописи (стенографии), знаком & Тирон заменял латинское «ЕТ», обозначающее *И* (по-английски And) Набор БЛЭ состоит из ряда компонентов, напоминающих детские кубики (см рис 57,а) Верхний ряд содержит БЛЭ, а нижний микросхемы для их реализации

Совет (**EWB**) Имейте в виду, что эти два ряда не связаны между собой ничем, кроме соответствия обозначений однотипных ЛЭ, поэтому частичное соответствие между УГО верхнего и нижнего ряда, начиная с третьего слева компонента, нарушается При выборе БЛЭ надо пользоваться их УГО, а не надписями, предназначенными для выбора микросхем

Картина будет иной (см рис 57,б), если при установке программы, по умолчанию было отмечено не DIN, как на рис 1, а – ANSI Тогда в работе окажется интерфейс стандарта ANSI (North



а)



б)

Рис 57 Наборы БЛЭ в разных стандартах а – DIN, б – ANSI (**EWB**)

American Standart – североамериканский стандарт, точнее American National Standart Institute – Американский национальный институт стандартов) В нем связь с древними римлянами напрочь исчезает, и УГО БЛЭ *И* скорее напоминает ядерную боеголовку современной баллистической ракеты (да и обозначения остальных ЛЭ, кроме операционного усилителя, не менее замысловаты)

Перечислим БЛЭ (слева направо)
2-Input AND Gate – двухвходовый ЛЭ *И*,
2-Input OR Gate – двухвходовый ЛЭ *ИЛИ*,
NOT Gate – инвертор (ЛЭ *НЕ)*,
2-Input NOR Gate – двухвходовый ЛЭ *ИЛИ-НЕ*,
2-Input NAND Gate – двухвходовый ЛЭ *И-НЕ*,
2-Input XOR Gate – двухвходовый ЛЭ *ИСКЛЮЧАЮЩЕЕ ИЛИ*,
2-Input XNOR Gate – двухвходовый ЛЭ *ИСКЛЮЧАЮЩЕЕ ИЛИ-НЕ*,
Tristate Buffer – буфер с тремя состояниями выхода (см выше рис 41),
Buffer – буфер,
Schmitte-Triggerred Inverter – инвертирующий триггер Шмитта (см далее в разделе «3 1 Триггеры»)

Рассмотрим свойства основных типовых БЛЭ. Из меню Logic Gates буксируем на рабочее поле 2-Input AND Gate – двухвходовый ЛЭ И (рис. 58,а).

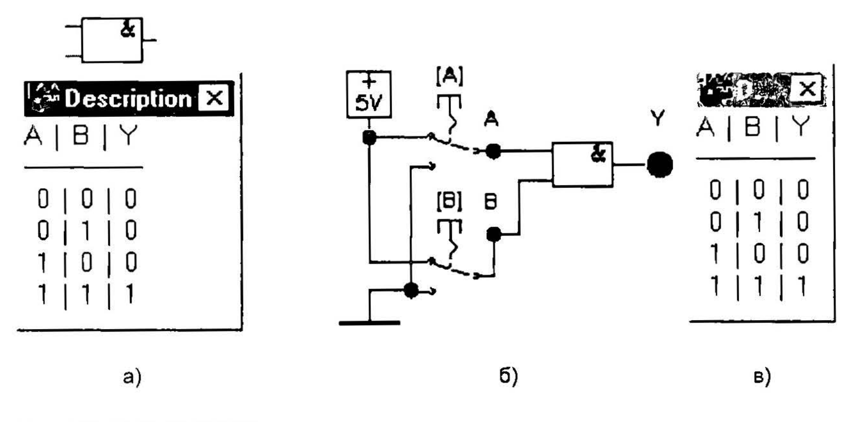


Рис 58 БЛЭ *И* (**EWB**)

Щелкаем ПКМ на его УГО и затем в открывшемся подменю ЛКМ по позиции Help. На экране возникнет окно предметной помощи: в данном случае надпись AND gate, сопровождаемая УГО этого элемента в стандарте ANSI, и соответствующая таблица истинности. Выделив курсором ТИ и выбрав в этом окне Правка>Копировать, закрываем эту американскую помощь и, открыв в основном окне Window и далее Description, вставляем туда (Past) ТИ. После сжатия этого окошка «вывешиваем» его в удобном месте рабочего поля (см рис. 58,а).

Совет (**EWB**) Описанную выше процедуру удобно выполнять для ознакомления со всеми новыми компонентами схем.

Проведем эксперимент с ЛЭ *И* (по-другому: &, And). Войдя в группу Sources, выберем источник постоянного напряжения +5 V (+Vcc Voltage Sources) . Этот источник, как и многие другие схемные виртуальные логические компоненты, имеет один вывод (второй – «земля» предполагается уже соединенным программным путем) Далее выбираем два переключателя, логический индикатор и собираем схему, показанную на рис. 58,б.

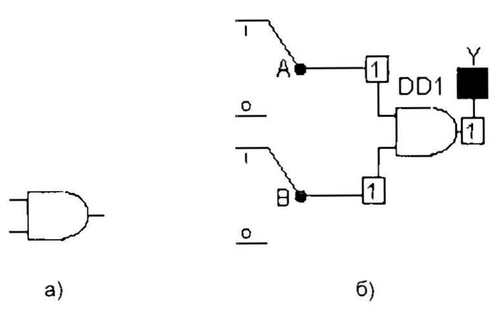
Совет (**EWB**) Не забудьте переименовать управляющие клавиши переключателей, заменив Space на А и В соответственно (иначе они будут срабатывать одновременно и только при нажатии на Space)

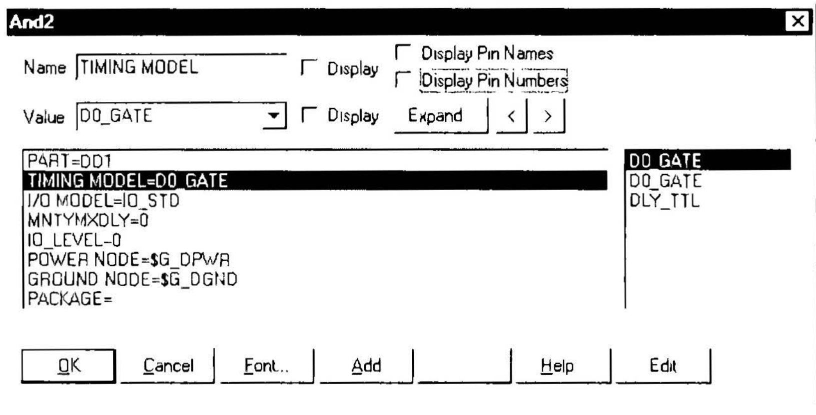
Включив моделирование и нажимая на клавиши А и В, задаем на соответствующих входах ЛЭ все возможные комбинации логических состояний входных сигналов Наблюдая за реакцией выходного индикатора и сверяясь с ТИ, убеждаемся, что ЛЭ & дает на выходе лог 1 (Y=1 – индикатор горит) только в одном единственном случае: на входе А=1 *И* на входе В=1 (здесь *И* играет роль соединительного союза, раскрывающего логический смысл этого ЛЭ) Внутри БЛЭ *И* может предполагаться любая из конкретных схем, осуществляющих эту операцию: ключевая – рис. 17,а, диодно-резистивная логика – рис. 21,а и т.п. Реальные устройства, разумеется, по ряду параметров будут отличаться от этих идеальных моделей.

В программе **MC** БЛЭ выбираются последовательностью команд Component >Digital Primitives>Standard Gates и далее в открывшемся списке тип ЛЭ Выбираем And Gates (ЛЭ *И*), а затем And2 (двухвходовый ЛЭ) На экране появляется УГО ЛЭ &, но в стандарте ANSI (см рис 59,а)

Конечно, можно было бы перерисовать это УГО. В программе **MC** есть такая возможность, и мы использовали ее для придания некоторым компонентам более удобного вида (например, для резистора), но в данном случае сочли это нецелесообразным. Дело в том, что эти УГО сплошь и рядом используются в переводной литературе и иных источниках, поэтому лучше к ним привыкать заранее. Это такая же напасть нашего времени (или бремени), как английский язык или $: любишь, не любишь, а приходится использовать.

После выбора компонента, его свойства редактируются в соответствующем окне (см. рис. 59,в). Здесь ограничиваемся выбором модели DO_GATE позиционного обозначения DD1. Далее выбираем два цифровых ключа и логический индикатор, из которых строим схему испытаний ЛЭ (рис. 59,б). Как и при испытаниях транзисторных логических схем, для выполнения моделирования даем команды: Analysis>Transient>Run и в схемном окне активизируем Node Voltages [13]. Перебирая варианты ключами А и В, убеждаемся, что на выходе будет лог 1 (Y=1 – индикатор горит) только в одном единственном случае, когда на входе А=1 *И* на входе В=1





в)

Рис 59 БЛЭ *И* (**МС**)

Совет (**МС**) Для того чтобы получить новые значения логических состояний цифровых узлов после изменения входных сигналов, надо каждый раз заново давать команду Run и в схемном окне активизировать Node Voltages.

Следующий ЛЭ *ИЛИ* также выбирается в программе **EWB** из меню Logic Gates Для этого на рабочее поле буксируется 2-Input OR Gate – двухвходовый ЛЭ *ИЛИ* (рис 60,а)

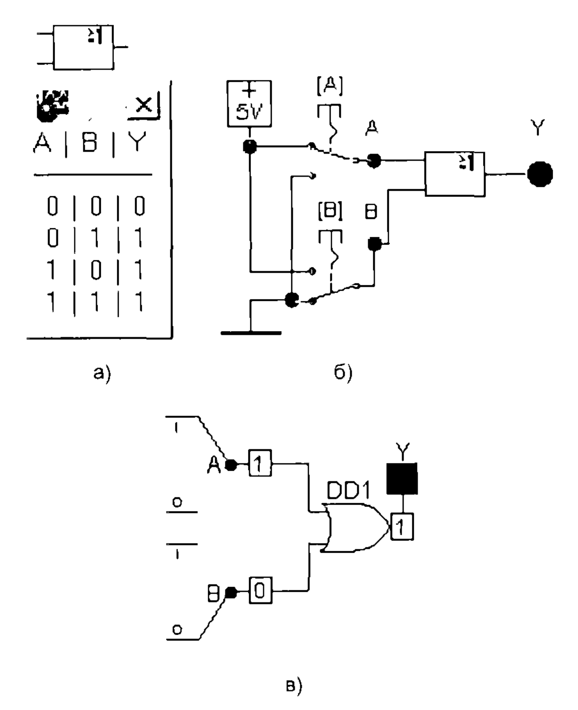


Рис 60 БЛЭ *ИЛИ* а, б – **EWB**, в – **МС**

Его ТИ показана на этом же рисунке Схемное обозначение ≥1 связано с тем, что 1 на выходе получается тогда, когда на входах есть хотя бы одна 1 Скопировав предыдущий файл и заменив в нем ЛЭ AND на OR, получаем схему, показанную на рис 60,б Проведя испытание этой схемы, убеждаемся в соответствии ее работы ТИ для ЛЭ *ИЛИ*

В программе **МС** ЛЭ *ИЛИ* выбираются командами Component >Digital Primitives>Standard Gates>OR Gates>OR2 (двухвходовой ЛЭ *ИЛИ*) На экране вновь появляется УГО ЛЭ в стандарте ANSI (см рис 60,в) Здесь по поводу американских форм УГО, как говорится, «без комментариев» Составляем схему, которую в этой программе также можно получить из предыдущей заменой ЛЭ AND на OR, и проводим ее моделирование (рис 60, в)

Далее выбираем инвертор В программе **EWB** на рабочее поле буксируется пиктограмма [1] NOT Gate – ЛЭ *НЕ* (рис 61,а)

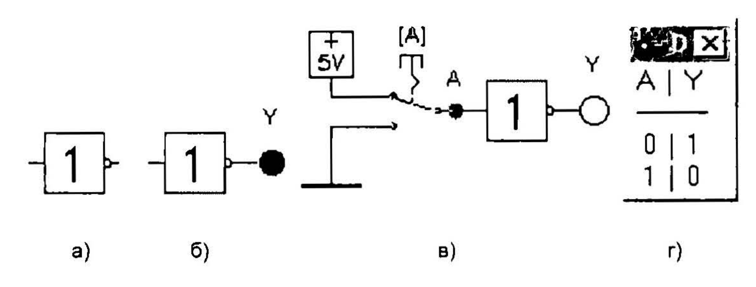


Рис 61 БЛЭ *НЕ* (**EWB**)

Этот ЛЭ имеет только один вход Его схемное обозначение на выходе имеет характерный кружок – символ инверсии. на выходе 0 (кружок), когда на входе 1(единица), изображенная внутри поля ЛЭ Присоединим к выходу этого виртуального инвертора индикатор и включим моделирование О, чудо – лампа горит! (см рис 61,б), хотя какие-либо сигналы отсутствуют Все дело в том, что в данной программе отсутствие входных сигналов автоматически считается за сигнал лог 0

Для реальных микросхем это совсем не так (хотя и там возникают проблемы), поэтому соберем нормальную схему (рис. 61,в) и, проведя ее испытание, получим ТИ инвертора, показанную на рис 61,г Из предыдущего случая надо извлечь тот урок, что случайные отсутствия соединений входов ЛЭ программа **EWB** будет считать как ни в чем не бывало и выдавать ошибочные результаты за достоверные Поэтому надо быть внимательными при виртуальных экспериментах

В программе **МС** ЛЭ *НЕ* выбираются командами: Component >Digital Primitives>Standard Gates>Inverters>Inverter (инвертор).

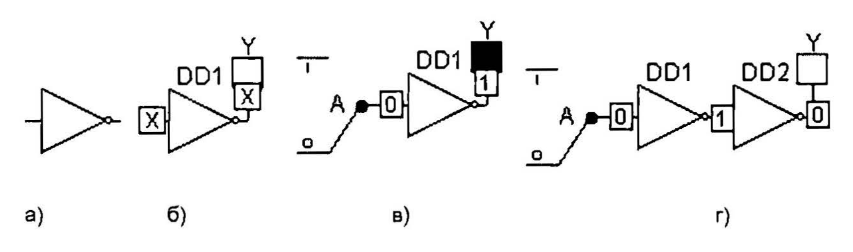


Рис 62 БЛЭ *НЕ* (**МС**)

УГО этого ЛЭ в стандарте ANSI (см рис. 62,а) – усилитель с инверсным выходом (кружок). Разумное обозначение достаточно вспомнить, что одиночный усилительный каскад на транзисторе дает на выходе инверсный сигнал (рис. 23,а и 49,б) Присоединяем к выходу индикатор и пытаемся повторить прежний фокус: на вход ничего не давать Проводим моделирование и получаем на входе X и на выходе X, т е неопределенному входу – соответствует неопределенный выход Это корректный результат Составляем схему с ключом на входе и проводим ее моделирование при A=0,Y=1 (рис 62, в) Если же пропустить сигнал через два последовательно включенных инвертора (или любое их четное число, включая, естественно, нуль, как число четное по соглашению), то он не изменит свой логический уровень (см рис 62,г) Этим свойством двойной инверсии широко пользуются при тождественных преобразованиях ЛФ

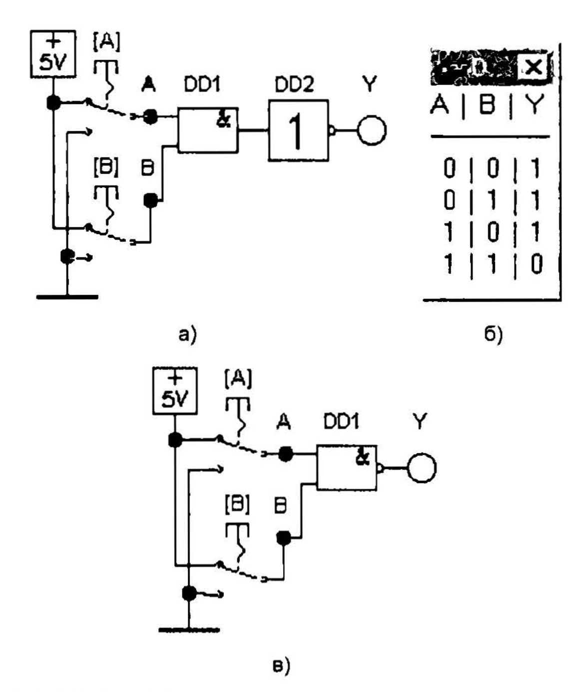
Комбинационная логика

Обладая тремя БЛЭ (*И, ИЛИ, НЕ*) можно, соединяя их между собой по определенным правилам, получить составные (комбинационные) ЛЭ, выполняющие более сложные ЛФ. Если при этом не использовать обратных связей и временных задержек, то получатся различные комбинационные устройства

Начнем с простейших комбинаций, которые уже были рассмотрены выше в различной элементной базе.

На выходе ЛЭ *И* поставим инвертор *НЕ* Собираем в программе **EWB** схему из двух БЛЭ DD1 AND и DD2 NOT (см. рис 63,а).

Проведя эксперименты с этой схемой, видим, что Y=0 только в одном случае: A=1, B=1 Подобный элемент, как уже указывалось, называется *И-НЕ*, на английском – NAND. Этот ЛЭ в «собранном» виде имеется в наборе БЛЭ и выбирается по пиктограмме с его УГО [&] 2-Input NAND Gate – двухвходовый ЛЭ *И-НЕ* (рис 60,б) Здесь к выходу ЛЭ & дорисован кружок, обозначающий инверсию сигнала Заменив в схеме рис 63,а ЛЭ DD1 AND и DD2 NOT на DD1 NAND, получаем схему, показанную на рис 63,б Испытания этого элемента дают ТИ, показанную на рис 63, в Сопоставляя эту таблицу с таблицей для ЛЭ AND, показанной на рис 60,а, видим, что они связаны инверсией выходного столбца Y.

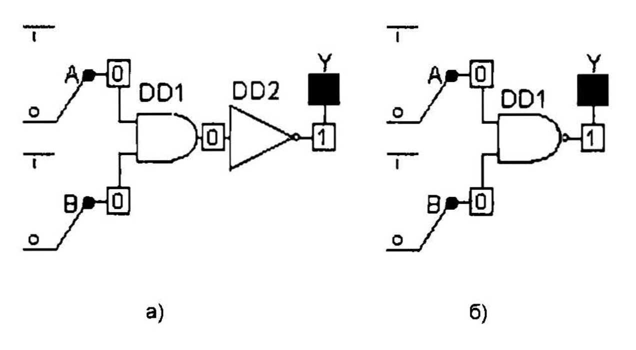


а) б)

в)

Рис. 63. БЛЭ *И-НЕ* (**EWB**)

Составной ЛЭ *И-НЕ* и его схемный эквивалент из библиотеки цифровых примитивов Nand2 в программе **МС** показаны на рис 64,а,б.



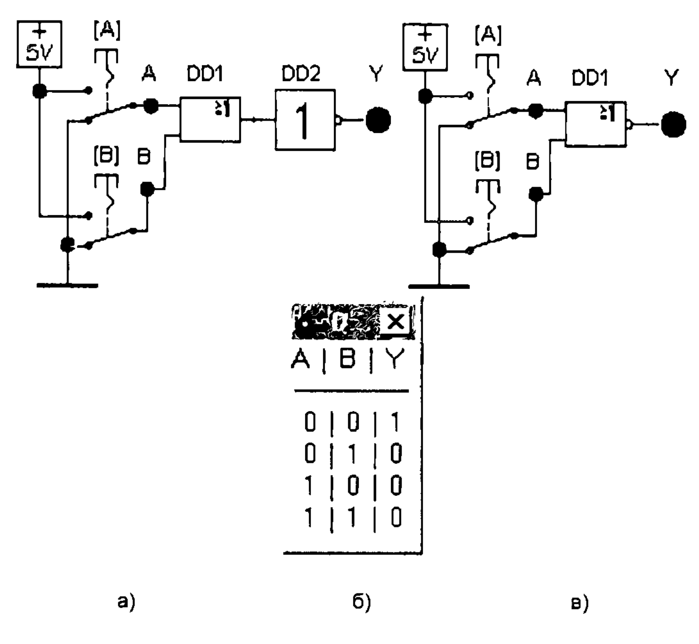
а) б)

Рис 64 БЛЭ *И-НЕ* (**МС**)

Здесь для обозначения инверсии выходного сигнала использована та же символика – кружок на выходе ЛЭ. Что ж, это удобная и экономная графическая форма, одна беда – слишком они малы и не всегда видны

Совет Будьте внимательны к этим «инверсионным» кружкам пропустив их или добавив лишние, можно получить обратный (ошибочный) результат

Теперь проделаем аналогичные операции, только используя ЛЭ *ИЛИ* Для этого в предыдущих схемах (рис 63,а и 64,а) заменим ЛЭ DD1 AND на ЛЭ DD1 OR и соответственно на рис. 63,б и 64,б ЛЭ DD1 NAND на ЛЭ DD1 NOR. В результате получим схемы для ЛЭ *И-НЕ*, по-английски NOR (см рис 65,66)



A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Рис 65 БЛЭ *ИЛИ-НЕ* (**EWB**)

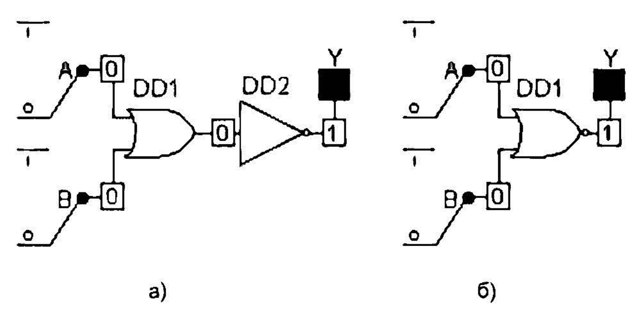


Рис 66 БЛЭ *ИЛИ-НЕ* (**MC**)

ТИ этого ЛЭ (см рис 65,в) дает Y=1 только при нулевых сигналах на входе (A=0, B=0) и является инверсной по выходному столбцу Y к ТИ, показанной на рис 60,а В программе **EWB** ЛЭ NOR, будто бы не включенный по входам, даст на своем выходе **1**, что может привести к сбоям в работе моделей устройств

Совет При сборке схем, особенно в программе **EWB**, отдельные компоненты могут касаться друг друга выводами, а соединение (виртуальная «спайка») будет отсутствовать, поэтому старайтесь избегать непосредственных соединений и используйте дополнительные соединительные провода или монтажные узлы В программе **MC** используйте контроль целостности соединений

Рассматривая ТИ ЛЭ NAND и NOR или, как уже говорилось, элементов Шеффера и Пирса, видим, что при A=B, т е в двух случаях (нулевых или единичных сигналов), оба эти элемента превращаются в обыкновенный инвертор Это можно легко проверить в виртуальном эксперименте, показанном на рис 67

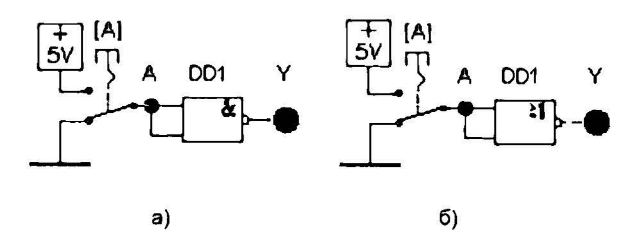


Рис 67 (начало)

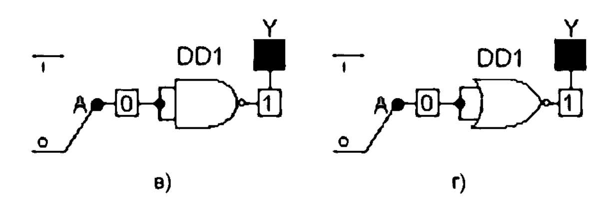


Рис 67 (окончание) Инвертор из БЛЭ Шеффера и Пирса
а, б – **EWB**, в, г – **МС**

Теперь решим одну практическую задачу, на которой покажем последовательность разработки комбинационных логических схем и, заодно, познакомимся еще с одним полезным виртуальным прибором

Задача о мажоритарном элементе (начало)

1 Словесное описание задачи

Часто при выборе альтернативных решений используется процедура голосования, при которой «побеждает большинство» Поскольку должен быть перевес хотя бы в один голос, то, отбрасывая авторитарный случай (единоличные решения) и считая голосующих равноправными, получаем, что минимальное число лиц, принимающих решение, например судей в коллегии, должно быть равно трем В логике соответствующая функция, принимающая истинное значение только тогда, когда большинство (для трех – это два или три) принимает истинное значение, называется функцией большинства, или мажоритарной (от лат major – больший, старший, в армии майор – воинское звание старшего офицерского состава)

Необходимо разработать такой ЛЭ, который давал бы сигнал на выходе, если два или три арбитра нажмут на кнопки «Да»

2 Формализация задачи

Присвоим трем судьям логические переменные А, В, С (порядок не дает приоритета) Каждая переменная может принять только два значения 0 или 1 Всего возможно 2^3 =8 вариантов или различных наборов аргументов в искомой ЛФ Y=F(A, B, C) Y=1, если любые два или все три аргумента равны 1 Необходимо найти вид этой функции и ее схемную реализацию

3 Решение

а) Составим таблицу истинности в соответствии с принятой формализацией Видоизменим ТИ по рис 38,б, отсекая восемь вариантов, убрав лишнюю переменную D и поменяв порядок следования переменных Столбец Y заполним заново, исходя из смысла задачи Так, нулевая строка (счет ведем с нуля) означает, что все трое судей против (0 , 0, 0), в этом случае и Y=0 Первая и вторая строки. два судьи «против», один «за» (0, 0, 1) и (0, 1, 0), значит опять Y=0 Наконец, в третьей строке один – «против», два – «за» (0, 1, 1), значит Y=1 Этот результат повторится в пятой и шестой строках, а также в седьмой, где все трое «за»

Окончательно получим таблицу истинности, отражающую сущность данной задачи, в виде, показанном на рис 68

№	A	B	C	Y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	**1**
4	1	0	0	0
5	1	0	1	**1**
6	1	1	0	**1**
7	1	1	1	**1**

Рис 68 Таблица истинности мажоритарного элемента

б) По таблице истинности составим ЛФ Искомая функция будет равна сумме произведений аргументов в строках ТИ, где Y=1, причем соответствующие сомножители берутся без инверсии, когда переменная равна 1, и с инверсией (отмечаемой штрихом `) при 0 Первая 1 появляется в строке №3. Здесь A=0, B=1 и C=1, следовательно, первое слагаемое состоит из сомножителей A`BC, строки №5, 6, 7 дают соответственно вклад AB`C, ABC`, ABC Окончательно, ЛФ примет вид·

Y= A`BC+ AB`C+ ABC`+ABC

(Следует отметить, что в ТИ допускается простановка неопределенных состояний знаком креста X Эти состояния являются промежуточными и нестабильными: в них ЛФ может принять значение 0 или 1)

в) Преобразование и минимизация ЛФ. Прежде чем находить необходимое схемотехническое решение задачи, желательно преобразовать ЛФ к определенному логическому базису, например, чтобы она выполнялась только элементами Шеффера. Кроме того, ее надо минимизировать, т е свести к минимуму по некоторому критерию, например числу входящих компонентов Эти процедуры должны выполняться по правилам булевой алгебры, с сохранением тождественности результата

Существуют различные способы решения подобных задач аналитический, с помощью специальных карт Карно и т д Поскольку мы работаем на компьютере, то и воспользуемся его возможностями Для этих целей в программе **EWB** разработан специальный уникальный инструментарий, поэтому прервемся на некоторое время в решении конкретной задачи и ознакомимся с ним.

Логический конвертор

В программе **EWB** войдем в панель Instruments (измерительные приборы) и в ней выберем Logic Analyzer или Logic Converter (логический конвертор – ЛК). На рабочем поле возникнет свернутое изображение прибора, которое служит для подключения к нему испытуемых схем ЛЭ Щелкнув по нему дважды ЛКМ, выведем на рабочее поле развернутую лицевую панель ЛК (см. рис. 69).

Этот виртуальный прибор, как ни странно, практически не имеет аналогов среди реальных физических приборов. На развернутой лицевой панели показаны восемь возможных аргументов ЛФ (входов ЛЭ) от А до Н и выход Out. Эти изображения не активны в схемном отношении, но они соответствуют клеммам, к которым подключаются входы и выход на схемном изображении, сами же буквы входят в ЛФ Одновременно кружки над буквами служат экранными кнопками, при нажатии на которые ЛКМ выбираются переменные ЛФ. Варианты, производимых преобразований (Conversions) изображены в виде символических пиктограмм-клавиш в правой части ЛК (см рис 69) Команда на эти преобразования дается щелчком ЛКМ по соответствующей клавише Результаты отображаются в двух специальных окошках. в главном – таблица истинности, а в нижнем – математическое выражение ЛФ. Кроме того, для ввода и вывода схем ЛЭ используется основное рабочее поле

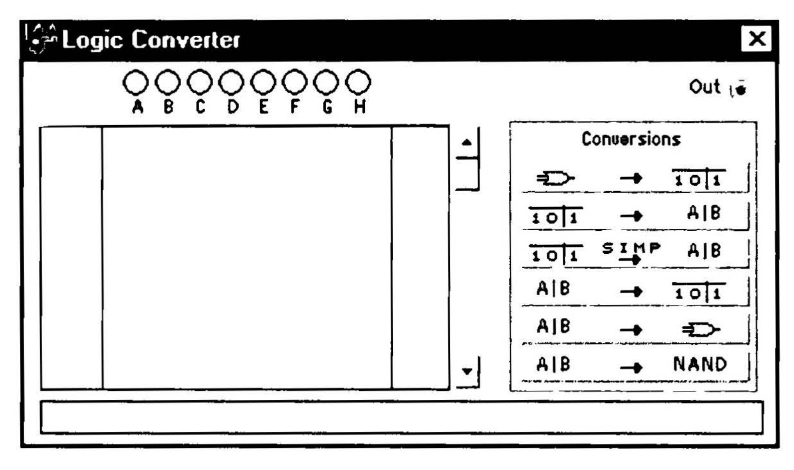


Рис 69 Лицевая панель логического конвертора (**EWB**)

Задача о мажоритарном элементе (окончание)

Вернемся к решению нашей задачи

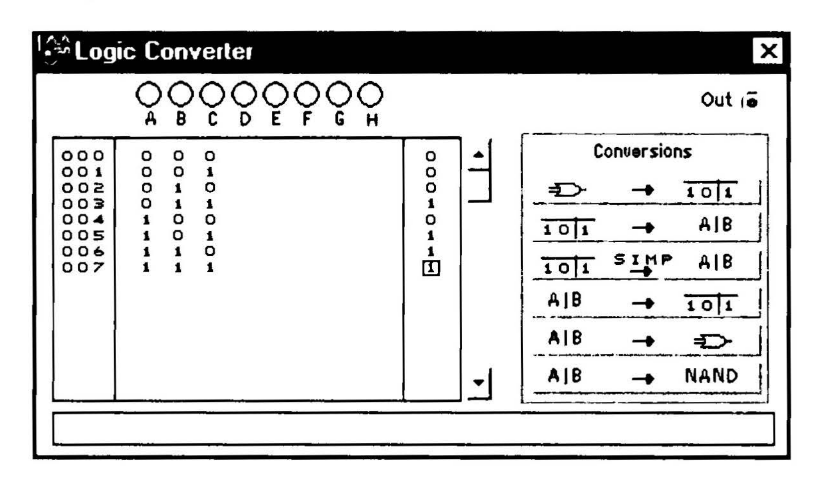


Рис 70 Лицевая панель ЛК с введенной ТИ (**EWB**)

Нажмем ЛКМ последовательно на кнопки А, В, С на панели ЛК (рис 70) (Повторное нажатие, как обычно, отменяет введенную команду) Теперь устанавливаем курсор в выходной (самой

правой) колонке и изменяем там нули на единицы в соответствие с ТИ на рис 68 Таблица истинности нашей задачи оказывается введенной в ЛК (рис 70) Нажав ЛКМ на вторую сверху клавишу 1 0|1 → A|B, осуществляем переход от ТИ к ЛФ, которая отображается в нижней экранной строке прибора (см рис 71,а)

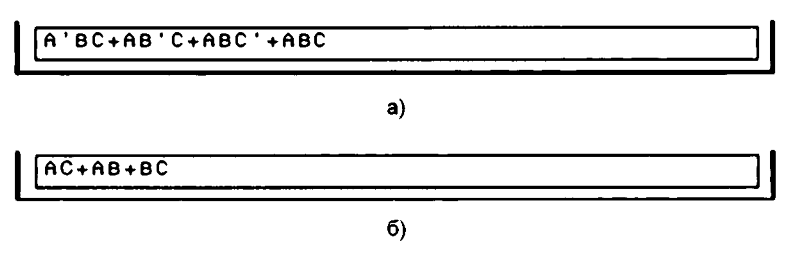


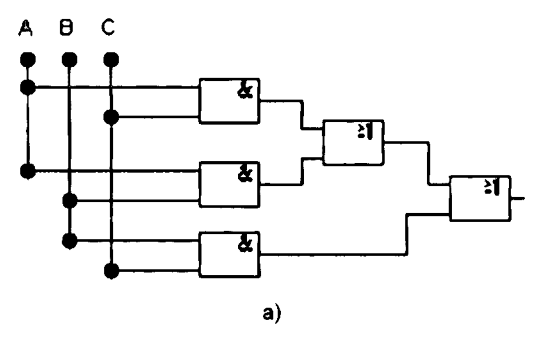
Рис 71 Экранная строка а – с первичной ЛФ, б – с упрощенной ЛФ (**EWB**)

Сопоставляя эту запись с приведенной выше, видим, что они полностью совпадают

Для упрощения ЛФ даем команду (Simplify – упростить), нажав ЛКМ на следующую клавишу ЛК 1 0|1 SIMP→ A|B. В результате получаем упрощенную ЛФ (см рис 71,б) в виде

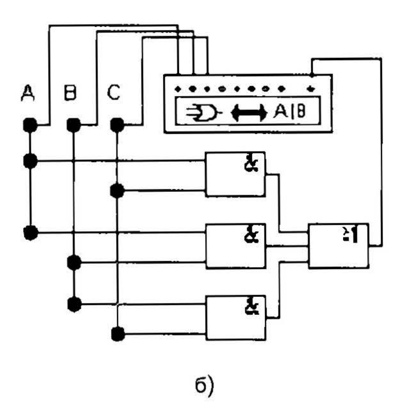
Y=AC+AB+BC

По этому выражению нажатием ЛКМ по предпоследней снизу клавише ЛК A|B → ⊐▷ строится искомая схема Она возникнет (рис 72,а) на основном экране в произвольном месте (возможно и позади панели ЛК) Панель ЛК надо временно свернуть



а)

Рис 72 (начало)



б)

Рис 72 (окончание) Проектирование мажоритарного ЛЭ (**EWB**)

Совет (**EWB**) Пока схема активна (выделена красным цветом), поместите на нее курсор (он превратится в изображение руки) и буксируйте ее на свободное место Это надо сделать обязательно, если новая схема наложилась на какие-либо старые «Растащить» после эти «завалы» практически невозможно, и если произошел сбой, лучше все начать сначала Не создавайте много схем на одном экране (в одном файле), работа с ЛК требует много оперативной памяти и могут начаться сбои

Проведем редактирование полученной схемы Заменим два двухвходовых *ИЛИ* на один трехвходовый ЛЭ, что соответствует структуре упрощенной ЛФ сложению трех произведений Дважды щелкаем по любому ЛЭ OR и в появившемся окне открываем позицию Number of Inputs (число входов) и отмечаем 3 На экране возникнет трехвходовый ЛЭ OR Лишний двухвходовый ЛЭ OR удаляем и проводим необходимое редактирование (рис 72,б) Кроме того, для проверки правильности схемы подключим ее к схемному компоненту ЛК (см рис 72,б) Откроем лицевую панель ЛК и если она имеет старое заполнение, то, нажимая на А, В, С и удалив ЛФ с экранной строки, подготавливаем прибор к работе

После этого, нажимая на первую клавишу, даем команду на преобразование «схема» → «таблица истинности» В результате должна появиться та же самая ТИ, а после нажатия на следующие две клавиши те же формулы

Таким образом, ЛК позволяет вводить исходную информацию через ТИ или схему, третий вариант заключается во вводе ЛФ через экранную строку с клавиатуры компьютера (правда, тут могут возникнуть небольшие проблемы с напечатанием штриха как знака инверсии, но они преодолимы) с последующим нажатием клавиши A|B → 1 0|1 . Все три способа представления ЦУ ТИ, схема и ЛФ взаимосвязаны и желательно уметь переходить от одного к другому

Уберем теперь ЛК и соберем, наконец, схему мажоритарного ЛЭ с ключами на входе и индикатором на выходе для ее исследования (рис 73,а) Аналогичную схему строим и в программе **МС** (рис 73,б) Проверив работу этих схем, убеждаемся в правильности решения задачи

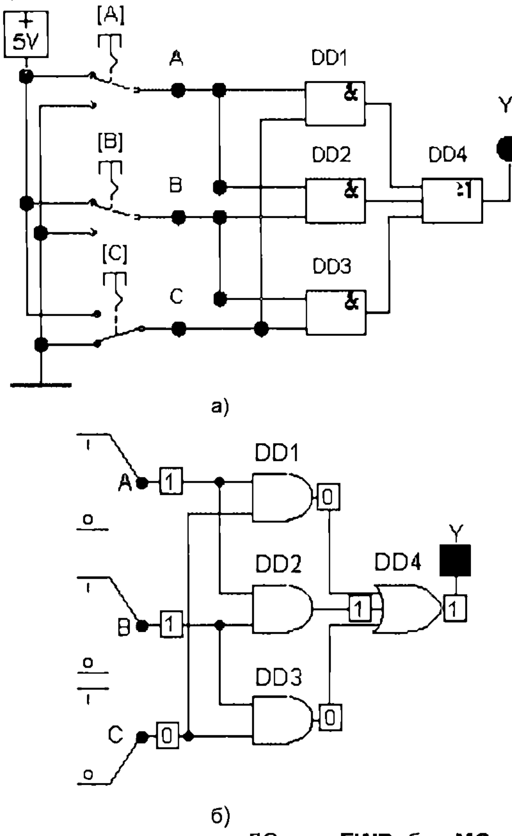


Рис 73 Исследование мажоритарного ЛЭ а – **EWB**, б – **МС**

Зададимся теперь целью сборки виртуальной модели мажоритарного ЛЭ на микросхемах. Для этого целесообразно перейти к однотипному базису БЛЭ, например NAND.

В программе **EWB** на лицевой панели логического конвертора после набора ЛФ (ТИ или подключения предыдущей схемы) нажимаем на нижнюю клавишу AIB → NAND и получаем соответствующую логическую структуру. Эта структура на рис 74 дополнена позиционными обозначениями БЛЭ DD1...DD6 и обозначением выхода Y.

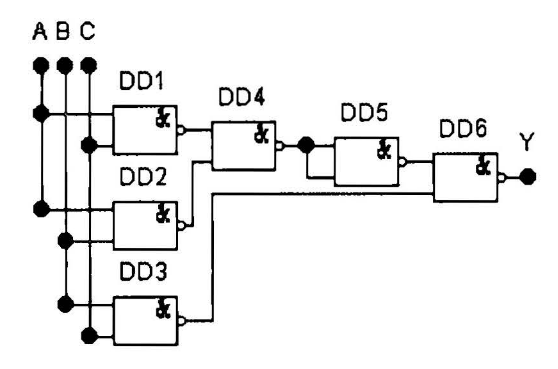
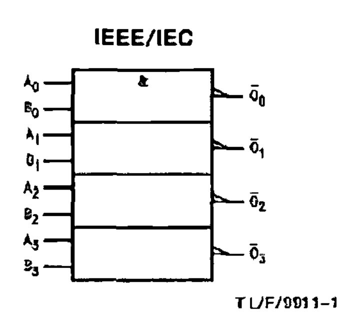


Рис 74 Логическая структура мажоритарного ЛЭ на БЛЭ NAND (**EWB**)

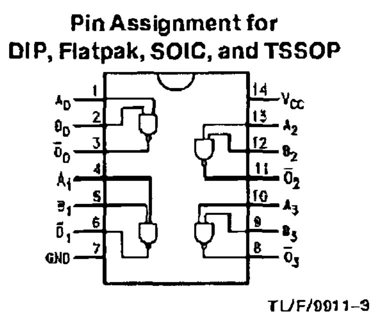
Итак, нам требуется микросхема с шестью двухвходовыми элементами NAND Обращаясь к справочникам и библиотеке компонентов, видим, что серию микросхем ТТЛ открывает микросхема 7400, содержащая в одном корпусе четыре необходимых БЛЭ Значит нам потребуется два таких компонента На рис. 75 приведена копия каталожных данных по этой микросхеме в части, касающейся схемотехники.

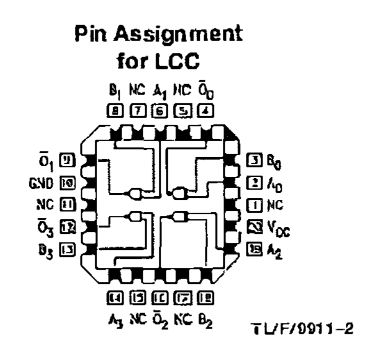
Все четыре отдельных логических элемента *2И-НЕ* данной микросхемы можно использовать независимо друг от друга. При подаче напряжения низкого уровня на один или оба входа каждого элемента на выходе устанавливается напряжение высокого уровня Если на оба входа подается напряжение высокого уровня, то на выходе формируется напряжение низкого уровня

Logic Symbol



Connection Diagrams





Pin Names	Description
A_n, B_n	Inputs
$\overline{O}_n$	Outputs

Рис 75 Интегральная микросхема 7400 фирмы National Semiconductor

В программе **EWB** микросхему 7400 можно выбрать, войдя через иконку (цифровые) в библиотеку цифровых микросхем Digital ICs и далее нажав на иконку . Можно также провести выбор через иконку Logic Gates (логические элементы), потом иконку и далее . Окончательный выбор после этих процедур производится по открывшемуся списочному меню Расположив на рабочем поле две микросхемы необходимо провести монтаж в соответствии со схемой, показанной на рис 74 Для этого надо разобраться с маркировкой выводов. Обозначения выводов элементов NAND следующие A и B – входы, Y – выход, причем каждый элемент еще имеет свою цифровую метку Введем обозначения компонентов и их выводов на принципиальную схему, так как это обычно делают при монтаже реальных радиоэлектронных устройств (см рис. 76,а).

Совет (**EWB**) Поскольку графический редактор данной программы дает возможность располагать текстовые метки узлов только строго над ними, то для того чтобы избежать их наложения на другие графические элементы схем, находите соответствующее расположение самих узлов, проводников, соединенных с ними, и, наконец, выполняйте при печати метки необходимый отступ за счет введения пробелов

Сборку схемы начнем с питания к VCC (14) подключаем «+» источника постоянного напряжения, а GND (7) заземляем Учитывая схему, показанную на рис 74, проводим необходимые внутрисхемные соединения, напоминающие монтаж на плате (см рис. 76,б) Дополнив схему ключами (A, B, C) и выходным индикатором Y, проводим ее испытание. Для того чтобы приблизить виртуальную модель к реальным устройствам, заменим однополюсные элементы источник питания – на батарею E1 и логический индикатор – на светоизлучающий диод VD1 с токоограничивающим резистором R1 (см рис. 76,в) Приведенная схема уже может быть использована для разработки соответствующей печатной платы устройства

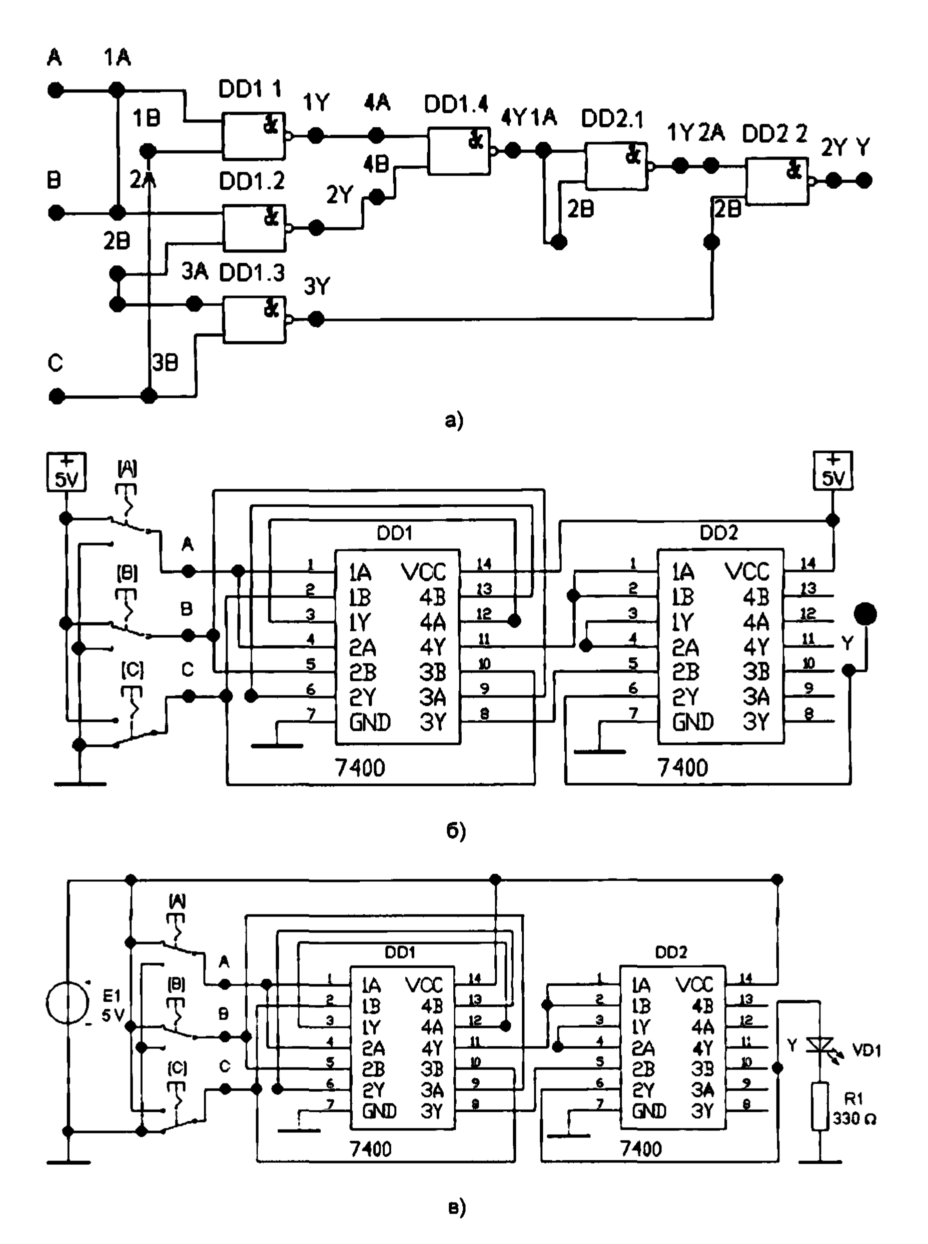


Рис. 76. Сборка мажоритарного ЛЭ на микросхеме 7400 (**EWB**)

Совет (EWB) Проводите виртуальный монтаж микросхем в большом масштабе изображения Пользуйтесь монтажными узлами для придания трассировке проводников необходимого расположения Используйте раскрашивание проводников для улучшения их распознавания

Программа **EWB** не ниже 5-й версии имеет дополнительный блок **EWB** Layout, в котором по полученной виртуальной монтажной схеме можно провести разработку всей печатной платы устройства Однако изложение этих вопросов далеко выходит за рамки данной книги

В программе **MC** отсутствует графическая корпусная эмуляция микросхем малой интеграции, хотя в ней и имеется обширная компонентная библиотека для использования их в схемотехнических моделях

Сложение двоичных чисел

Именно эта математическая операция по существу лежит в основе работы ЭВМ первые машины и называли арифметическими, а главным мозгом процессора является АЛУ (арифметическо-логическое устройство) Обычно школьникам вбивают в голову таблицу умножения Здесь же ее место по праву занимает «таблица сложения»

Сложение производится поразрядно (побитно) и, поскольку машина знает только нули и единицы, то результатом сложения двух единиц является 0 в данном разряде и знак *, которым обозначен перенос в следующий разряд Перенос *(по-английски – Carry) в соответствующем разряде складывается наряду с цифрами, стоящими в этом разряде Таким образом, таблица сложения двоичных чисел такова

0+0=0	* +0+0=1
0+1=1	* +0+1=0 и *
1+0=1	* +1+1=0 и *
1+1=0 и *	* +1+1=1 и *

Полусумматор

Построим схему полусумматора – ЦУ с двумя входами для сложения двоичных чисел без учета переноса из предыдущего

разряда (разумеется, перенос в следующий разряд должен формироваться) Значит, полусумматор реализует первую колонку этой таблицы, и он имеет два выхода сумму в данном разряде и перенос в следующий разряд Это было словесное описание задачи Теперь ее формализация обозначим складываемые биты через А и В, сумму – S (от лат Summa) и перенос С (от англ Carry) Тогда ТИ полусумматора, имеющего два входа и два выхода, будет такова

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Нетрудно видеть, что перенос формируется ЛФ *И*, т е С=АВ Для составления ЛФ S конечно можно обратиться к ЛК, но здесь такой простой случай, что результат, как говорят, на лицо и его надо лишь увидеть

Составляем, как было описано выше, ЛФ по выходным единицам их две Следовательно, будет два слагаемых

S=A’B+AB’

Полученная функция очень важна и имеет специальное название *Исключающее ИЛИ*, на английском – XOR (от eXclusive OR) Обычное *ИЛИ* можно в противовес назвать «включающим» *Исключающее ИЛИ* осуществляет математическую операцию сложение двух бит по модулю 2, поэтому иногда в обозначениях полусумматора можно увидеть M_2 Математики часто обозначают эту операцию специальным знаком ⊕ «псевдоплюс», т е

$S=A \oplus B$

Для составления логической структуры схемы суммирования в соответствии с формулой S=A’B+AB’ надо начинать с последнего действия Знаку «+» соответствует логическая операция *ИЛИ* Значит, на выходе ставим ЛЭ OR Умножению соответствует логическая операция *И* Здесь две таких операции, да еще один ЛЭ требуется для переноса Всего три ЛЭ AND Наконец, каждую переменную надо проинвертировать нужны два ЛЭ NOT Выводим эти шесть ЛЭ на рабочее поле и собираем схему, дополняя ее источниками сигналов и индикаторами (рис 77,а), и аналогично в программе **МС** (рис 77,б)

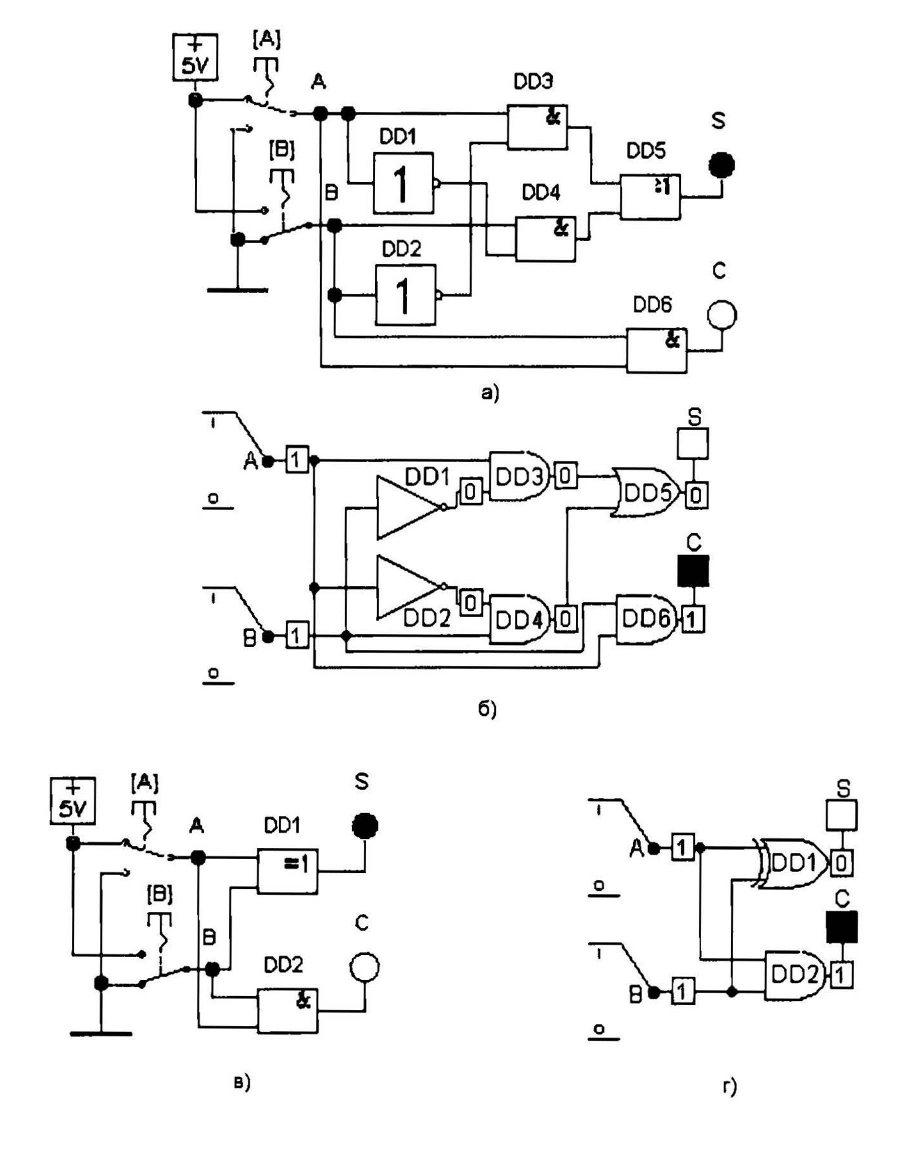


Рис. 77. Логические структуры полусумматора:
а – **EWB**; б – **MC**: в – **EWB**; г – **MC**

Результаты моделирования сверяем с алгеброй Буля, так сказать «проверяем алгеброй гармонию» виртуальных схем.

В рассмотренных схемах полусумматора подсхема (часть всей схемы) на ЛЭ DD1 DD5 представляет сборку ЛЭ *Исключающее ИЛИ*. Этот элемент имеется в нашем компьютерном конструкторе в виде целого ЛЭ Его текстовое обозначение XOR, а УГО в **EWB** имеет на своем поле знак булева (полужирным шрифтом) равенства единице =1. Это связано с тем, что ЛФ *Исключающее ИЛИ* равна 1 только тогда, когда какой-либо один аргумент равен 1 Специалисты выражают эту ЛФ фразой: «Что-нибудь, но не все». В программе **МС** (Североамериканский стандарт ANSI) УГО не поддается ни описанию, ни какому-либо рациональному объяснению. Рискнем предположить или пошутить, что здесь сказалась генетика тотемных знаков североамериканских индейцев Поэтому лучше один раз увидеть . Выберем ЛЭ XOR и, переделав предыдущие файлы, построим на нем полусумматор из двух ЛЭ (см рис. 78, а, б)

Если на выход ЛЭ *Исключающее ИЛИ* поставить инвертор, то получится *Исключающее ИЛИ-НЕ*, по-английски eXclusive Not OR (XNOR) ТИ элемента XNOR имеет на выходе Y 1, 0, 0, 1 УГО отличается от XOR в обозначениях ANSI наличием кружка инверсии на выходе, в обозначениях DIN программы **EWB** кружка нет, а в основном поле УГО ЛЭ убрана 1. Этот ЛЭ имеется в обеих программах

В программе **EWB** имеется специальный инструментарий для выделения функциональных блоков из больших схем, но его можно также использовать для создания дополнительной библиотеки цифровых узлов. Проведем построение субблока полусумматора по схеме на рис. 77,а. Прежде всего, путем простого растяжения выделенных частей, придадим схеме такой вид, чтобы окончательно внутрь выделяющей рамки входил только полусумматор (см рис. 78,а). Это и будет субблок (субцепь, подсхема, подцепь) Проводники, пересекающие контур субблока, впоследствии образуют его выводы, поэтому их число и расположение надо строго контролировать. После выделения субблока, пока его элементы имеют активный красный цвет, входим в меню Circuit (цепь) и выбираем Create Subcircuit (создание субцепи). В появившемся меню Subcircuit (см. рис 78,б) впечатываем в строке Name название, например НА (от – Half-Adder), и выбираем Copy from Circuit (копирование из цепи). На рабочем поле появится дополнительное окно с развернутой схемой субблока (рис 78,в)

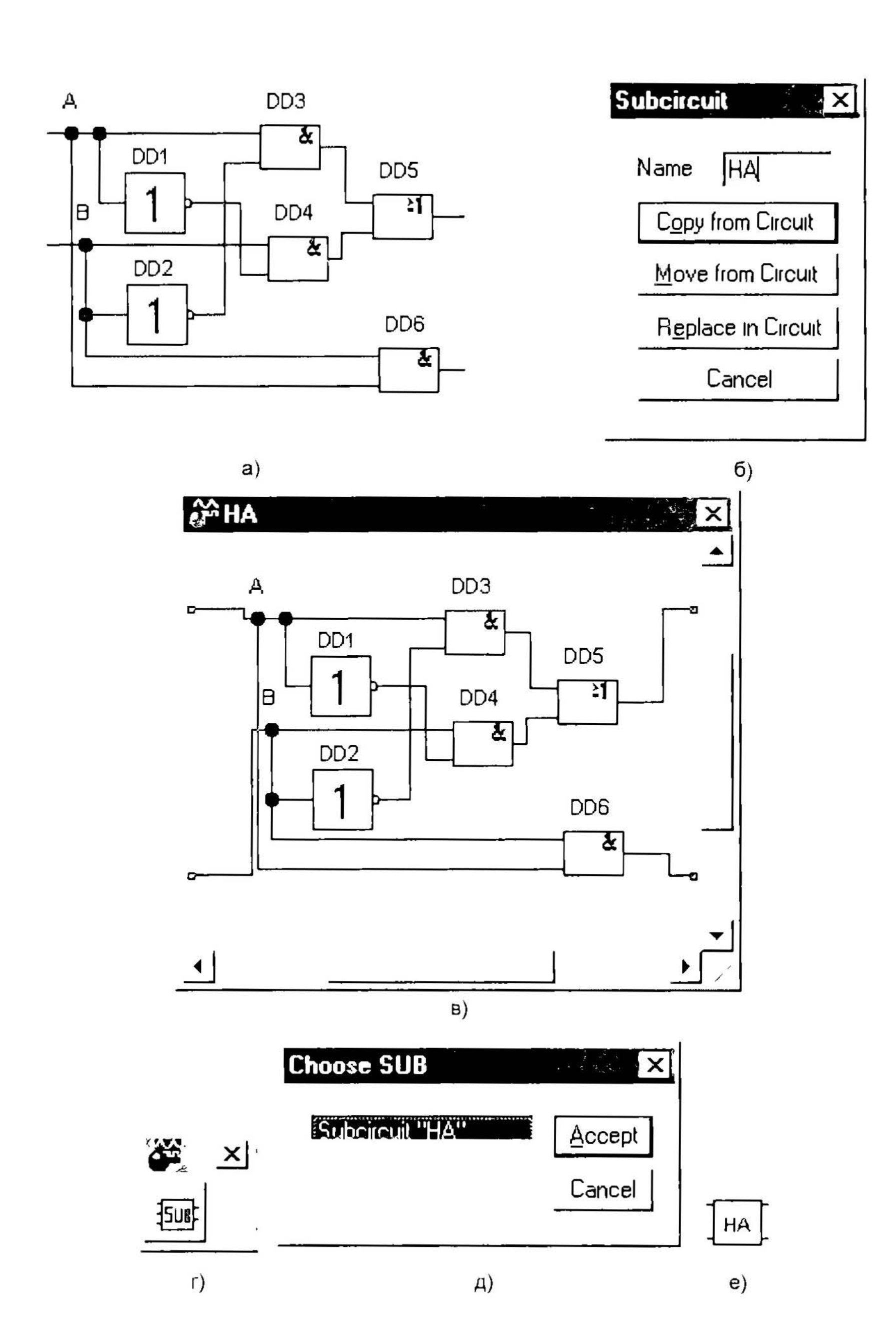


Рис 78 Создание субблока полусумматора (**EWB**)

При необходимости эту подсхему можно отредактировать стандартными приемами Кроме того, можно добавить выводы, вытянув провод из нужной точки до края внутреннего поля окна После отпускания ЛКМ возникнет дополнительная контактная площадка Напротив, если контактную площадку вытянуть за пределы окна, то соответствующий вывод будет удален После редактирования и проверки правильности выбранной подсхемы это окно можно закрыть

Совет (**EWB**) Пока подсхема активна, проверьте, какие элементы выделены красным цветом Обязательно проверьте правильность развернутой схемы полученного субблока Если после копирования в исходной схеме или копии ее субблока появятся графические искажения, то их необходимо исправить стандартными приемами

Для выбора схемного изображения субблока в свернутом виде, надо нажать в ряду выбора компонентов на иконку Favorites (избранные) В результате получим иконку с изображением субблоков (см рис 78,г) и возможность дальнейшего выбора Choose SUB (см рис 78,д) Нажав в последнем окне ЛКМ Accept (согласиться), получим искомое схемное изображение субблока в свернутом виде (рис 78, е) К «внутреннему содержанию» субблока всегда можно вернуться двойным щелчком ЛКМ по его схемному изображению

Попутно отметим, что библиотеку раздела Favorites можно доукомплектовать уже существующими и наиболее часто используемыми компонентами, нажав на соответствующую иконку ПКМ и выбрав опцию Add to favorites (добавить к избранным) Создание такой специализированной пользовательской панели для конкретной схемы иногда бывает удобно Для использования полученного субблока в новом окне, необходимо скопировать схемное изображение субблока в свернутом виде (рис 78, е) в буфер обмена Открыв новое окно (New) и нажав на иконку Favorites , увидим, что внутри нет никаких элементов Нажмем ЛКМ Past на экране возникнет схемное изображение субблока, а в открытом окошке – Favorites иконка для его выбора (как ранее на рис 78, г)

В программе **МС** также предусмотрена возможность как пополнения библиотек новыми или нестандартными компонентами,

так и разработки макромоделей различных устройств. Однако эти процедуры значительно сложнее и требуют дополнительного знания специальных языков для их описания. Эти процедуры описаны в литературе и здесь не рассматриваются.

В цифровом наборе Digital программы **EWB** существует специальный библиотечный компонент: полусумматор (по-английски – Half-Adder) . Данный компонент эквивалентен созданному нами субблоку, но имеет иное графическое оформление (изображение знака суммирования Σ, а также два входа и два выхода, один из которых снабжен пометой: CO – Carry Out (выход переноса). Кроме того, данный схемный компонент располагается в основной библиотеке, что, конечно же, удобнее Приведенный пример создания субблока был выполнен по двум соображениям: во-первых, показать саму технику создания новых по отношению к библиотечным субблоков и, во-вторых, раскрыть возможное «содержание» библиотечного полусумматора В отличие от вновь созданных субблоков готовые библитечные компоненты не раскрывают своих схем и не дают возможности их редактировать (в большинстве компонентов редактируются только их параметры).

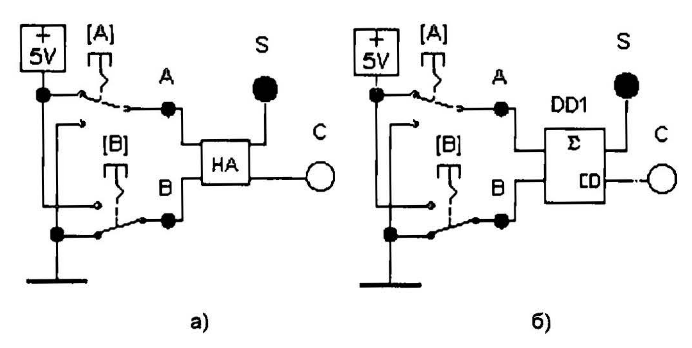


Рис. 79. Схема включения полусумматора (**EWB**)

Схемы, в которых использован вновь созданный и стандартный полусумматоры показаны на рис. 79,а,б. В программе **MC** подобный компонент отсутствует. Его при необходимости, конечно, можно «сделать» из имеющейся заготовки логического четырехполюсника (Logic2·2), предназначенного для записи логиче-

ских выражений, но это требует ознакомления с достаточно специальными форматами ведения подобных записей.

Последовательностная логика

Теперь нарушим то основное правило в сборке комбинационной схемы из БЛЭ, согласно которому ее выход не мог быть одновременно связан с каким-либо входом, составляющих ее ЛЭ В этом случае выходной сигнал возвращался бы обратно на вход, т е в схеме образовалась бы цепь обратной связи Теперь же направленно введем подобные цепи обратной связи

В программе **EWB** возьмем два элемента *ИЛИ-НЕ* (NOR Gate) и соберем на них простейшую схему (рис. 80,а), в которой сигнал с выхода Out1 ЛЭ DD1 подается на один из входов ЛЭ DD2 Как обычно, исследуем состояние выхода Out1 в зависимости от состояния входов A и B.

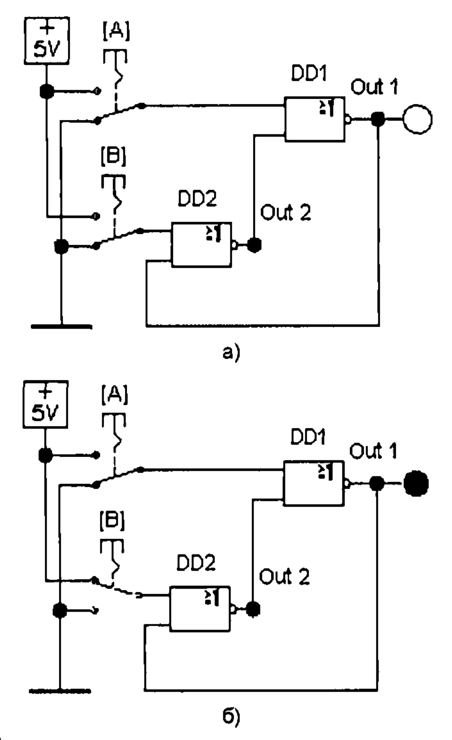


Рис 80 (начало)

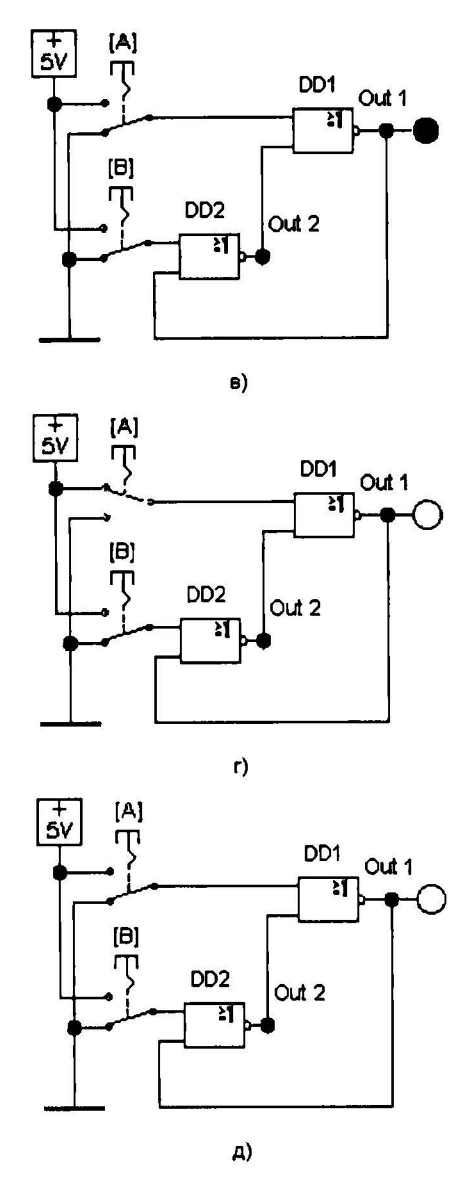
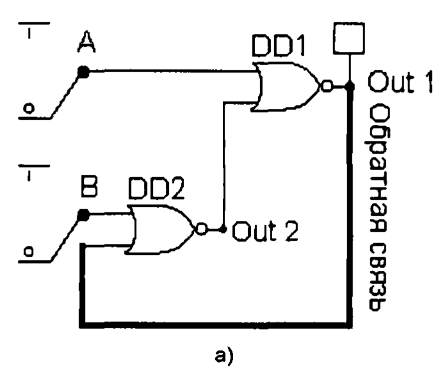


Рис 80 (окончание). Схема с обратной связью (**EWB**)

Если при A=0, B=0 включить моделирование, то индикатор на выходе будет мигать, что соответствует неопределенному состоянию схемы в этой программе. Подадим B=1, оставив A=0, и включим моделирование: получим Out1=1 (рис. 80,б) Затем, не выключая моделирование, вернемся к B=0. О, чудо! – индикатор продолжает гореть, т.е по-прежнему Out1=1 (рис. 80,в) Представьте себя на минуту на месте электрика, собравшего осветительную электропроводку с двумя выключателями и лампочкой Вот он выключил уже оба выключателя, а лампочка, как ни в чем не бывало, продолжает гореть! Нажимая многократно на ключ B (не выключая моделирование), видим, что эта клавиша как бы потеряла свои управляющие свойства после первого включения. Но это еще не все. Не выключая моделирование, после B=0 переведем A из нуля в единицу, т.е. выполним A=1. Тогда индикатор погаснет, т.е. Out1=0 (рис. 80,г). И, наконец (не выключая моделирование), выполнив A=0, вернемся к исходному состоянию, когда A=0, B=0 и Out1=0 (рис 80,д) Теперь, воображаемый электрик был бы вполне доволен, но, повторив все сначала, он бы снова поразился, получив те же результаты: при двух выключенных выключателях (A=0, B=0) лампа может как гореть (рис 80,в), так и не гореть (рис. 80,д).

Теперь обратимся к программе **MC**. Выбрав два БЛЭ NOR, два ключа Switch и логический пробник-индикатор LED, соберем из них аналогичную схему с обратной связью (рис. 81,а).

Задав A=0 и B=0, после анализа Transient, получим на выходе Out1=x – неопределенное состояние (рис. 81,б). Задав A=0 и B=1, после анализа Transient, получим на выходе Out1=1 (рис 81,в) Задав A=1 и B=0, после анализа Transient, получим на



а)

Рис. 81 (начало)

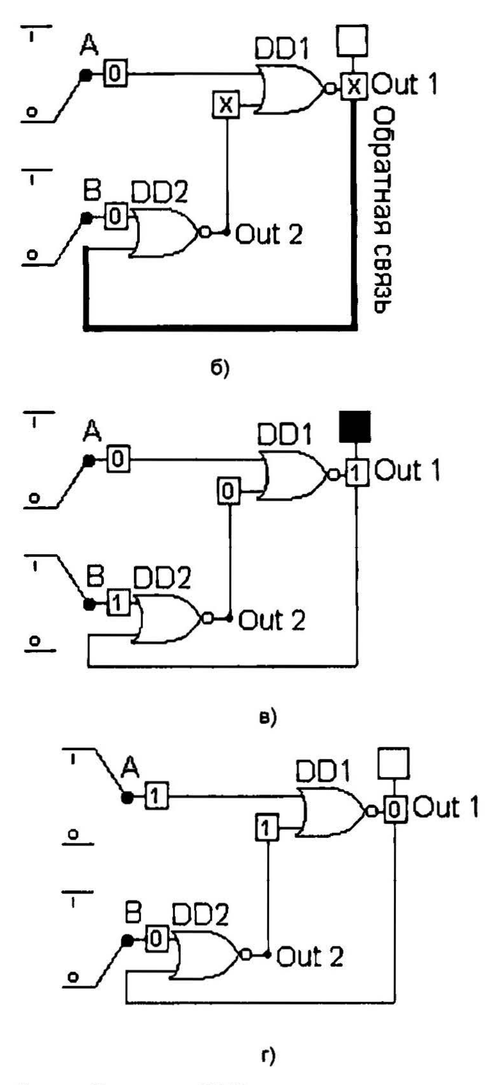


Рис 81 Схема с обратной связью (**MC**)

выходе Out1=0 (рис. 81,г). Никакие «чудеса» пока не наблюдаются Дело в том, что в данной программе моделирование проводится в анимационном режиме ключей из нулевого начального состояния, и нет возможности их переключения в процессе счета.

Поэтому заменим ключи специальными генераторами цифровых сигналов (Stimulus Generators).

Выбор генераторов производим, последовательно нажимая ЛКМ: Component>Digital Primitives>Stimulus Generators>Stim1. Последняя цифра 1 означает, что это генератор одноразрядного цифрового сигнала. В появившемся окне редактирования свойств (рис 82) присваиваем COMMAND=B, впечатав B внутри окошка Value и рядом отмечаем Show (см. рис 82).

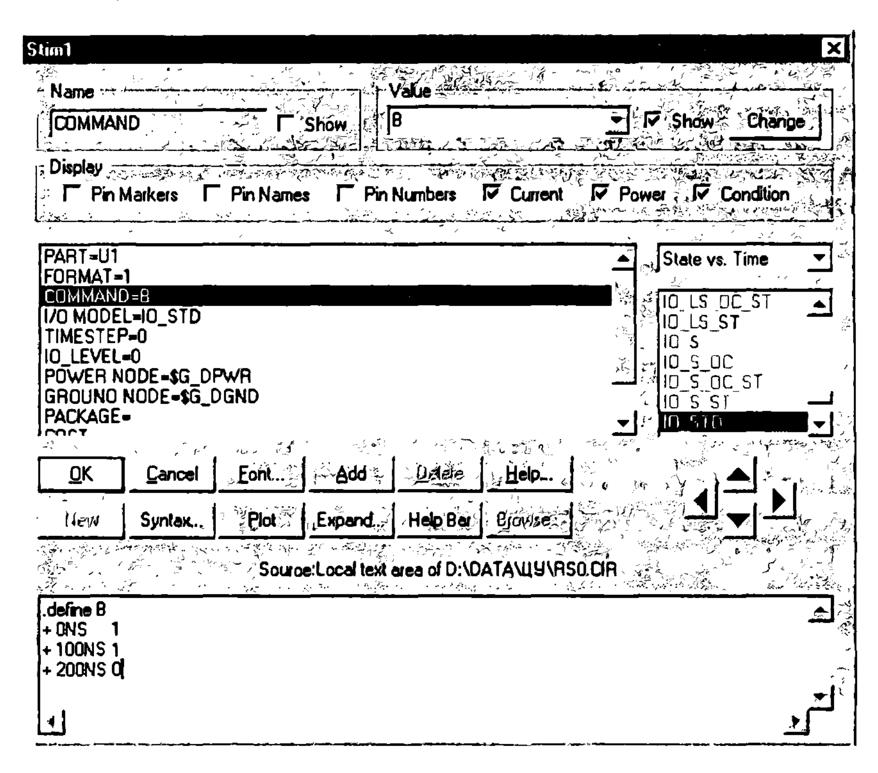


Рис 82 Редактирование свойств генератора цифрового сигнала (**MC**)

Здесь же, в самом нижнем окне, в специальном формате определяем источник B. Пусть полное время моделирования равно 200 нс (NS или ns) Разобьем его на два интервала по 100NS на первом интервале (от 0NS до 100NS) B=1, а на втором (от 100NS до 200NS) B=0 Таким образом, в нижнем окошке (см. рис. 82) напечатаем такое определение:

.define B
+0NS 1
+100NS 0
+200NS 0

Аналогично определим источник сигнала А, который пусть будет все время на нулевом уровне, и в соответствующем окошке напечатаем:

.define A
+0NS 0
+100NS 0
+200NS 0

Сменив цифровые ключи на новые источники, получим схему, показанную на рис. 83, где в прямоугольных рамках показана нумерация цифровых узлов d1–d4 (состояние индикатора выхода пока неинформативно).

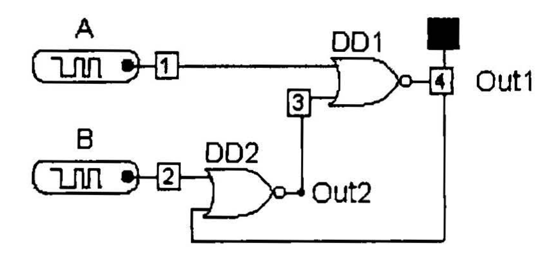


Рис. 83. Схема с генераторами цифрового сигнала (**МС**)

Transient Analysis Limits

Run | Add | Delete | Expand.. | Stepping.. | Properties. | Help..

Time Range: 200NS
Maximum Time Step: 0
Number of Points: 51
Temperature: Linear 27
Run Options: Normal
State Variables: Zero
☑ Operating Point
☐ Operating Point Only
☑ Auto Scale Ranges

P	X Expression	Y Expression	X Range	Y Range
1	T	d(1)	2e-7,0,4e-8	20
1	T	d(2)	2e-7,0,4e 8	20
1	T	d(3)	2e-7,0,4e-8	20
1	T	d(4)	2e-7,0,4e-8	20

Рис. 84. Установки режимов анализа (**МС**)

Входим в режим анализа и делаем соответствующие установки, показанные на рис. 84. Включив команду на моделирование (Run), получим на схеме (после нажатия на Node Voltages) состояние цифровых узлов d1–d4 к концу моделирования и индикатора выхода (рис. 85,а).

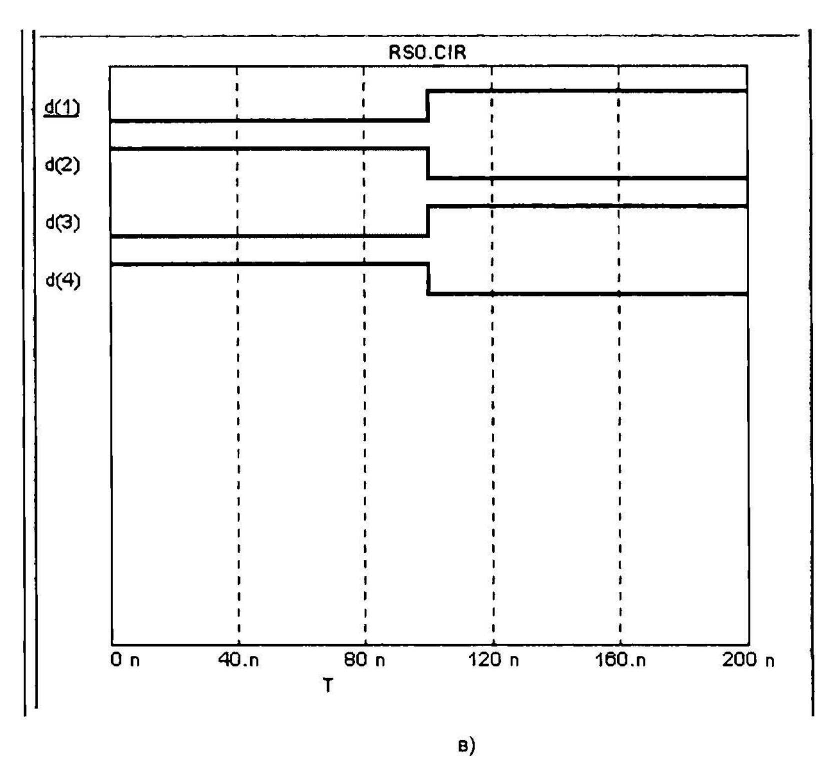
A
1:0
DD1
3 0
4 1
Out1
B
2 0
DD2
Out2

а)

RSO CIR
d(1)
d(2)
d(3)
d(4)
0.n
40.n
80.n
120.n
160.n
200.n
T

б)

Рис. 85 (начало)

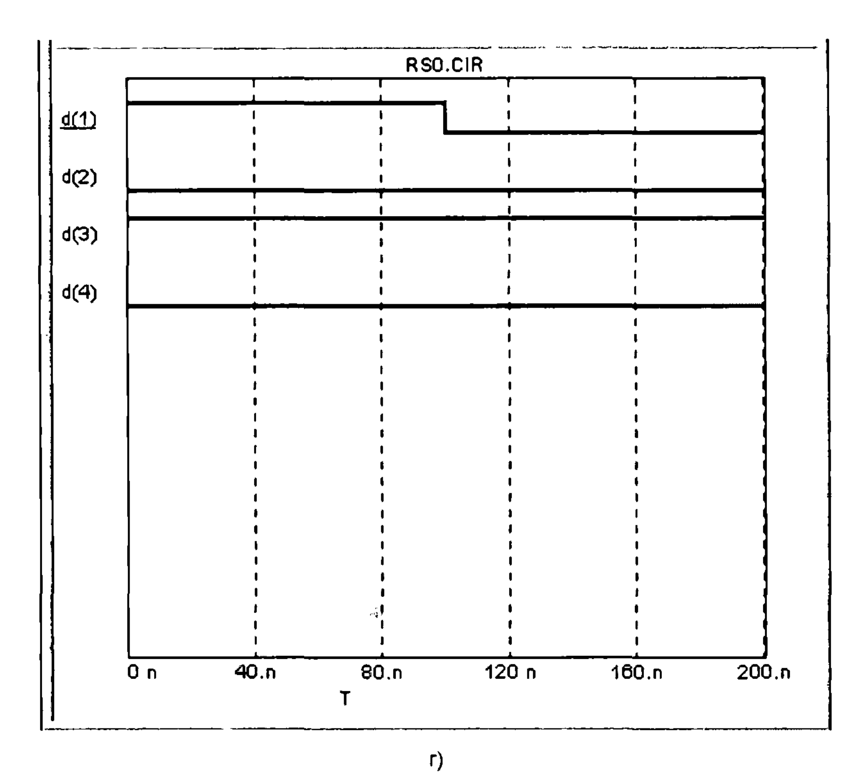


в)

Рис 85 (продолжение)

Кроме того, результаты анализа будут и на графиках (рис. 85,б), на которые мы до сих пор не обращали внимания Как нетрудно видеть, эта картина совпадает с результатом, показанным на рис. 80,в Вот и появилось первое «чудо» Устройство вроде бы выключено, а индикатор горит!

Теперь посмотрим, что будет, если после состояний A=0 и B=0, когда индикатор горел, сделать A=1 Для этого изменим значения входных цифровых сигналов во времени Это можно выполнить, открывая окна их свойств, но проще, перейдя из основного схемного окна (Page) в текстовое окно (Text), дублирующее окно свойств Соответствующие кнопки размещены в левом нижнем углу схемного окна



г)

Рис. 85 (окончание). Моделирование схемы с обратной связью (**МС**)

Выставим в текстовом окне следующие значения сигналов:

define A
+0NS 0
+100NS 1
+ 200NS 1

define B
+0NS 1
+100NS 0
+ 200NS 0

Вернувшись в основное окно, проведем моделирование и получим результат (см рис. 85,в), который по существу совпадает с результатом, показанным на рис. 80,г И, наконец, задав новые значения сигналов:

```
.define A
+0NS 1
+100NS 0
+ 200NS 0

.define B
+0NS 0
+100NS 0
+200NS 0
```

и выполнив моделирование (см рис 85,г), вернемся к исходному состоянию, аналогично рис 80,д

Основным отличием поведения рассмотренной схемы с обратной связью является то, что она запоминает предыдущие состояния и реагирует на новые команды не только по их значениям, но и с учетом этого Таким образом, удивительное поведение связано с временной последовательностью команд и памятью

Если в аналоговых электронных устройствах применение обратной связи позволяло в зависимости от ее знака, глубины и типа регулировать амплитудно-частотные характеристики, проводить температурную стабилизацию, переводить систему в режим автоколебаний и т.п , то в цифровых устройствах мы встречаемся с качественно новым эффектом обратной связи – памятью систем.

Подробнее эти вопросы будут рассмотрены в гл 3

После данного общего введения в цифровую электронику и моделирование ее простейших элементов, перейдем к более полному рассмотрению основных цифровых узлов и устройств

2. УЗЛЫ КОМБИНАЦИОННОГО ТИПА

2.1. Шифраторы и дешифраторы

Легран разогрел пергамент и дал его мне. Между черепом и козленком, грубо начертанным чем-то красным, стояли такие знаки
53 ╬╫╟+305)6*;4826)4╬.)4 ╬),806*;48+ 8‖))..
Эдгар По. Золотой жук

В цифровых устройствах используют специальные способы систематического представления информации – кодирование Развитие техники передачи информации, ее обработки и хранения привело к разработке самых различных кодов. Примерами кодов являются: знаменитая «морзянка» (азбука Морзе), радиолюбительский Q-код, штрих-код. По сути, представление чисел в различных системах счисления (двоичной, восьмеричной, десятичной, шестнадцатеричной и др.) является простейшим *числовым* кодированием. Коды используются для сжатия информации, обеспечения помехоустойчивости, защиты и т.п.

А скольким слепоглухонемым людям помог читать известный шрифт Брайля, являющийся по своей сути двоичным кодом, так как любая точка текста в нем имеет только два состояния: выпуклое или плоское. Широко распространенным кодом является код ASCII (American Standard Code for Information Interchange – стандартный американский код для обмена информацией). Именно в этом коде принято сохранять информацию в памяти персональных компьютеров. В двоичном коде производятся вычисления на компьютере. Кроме перечисленных, на практике используют коды: позиционный, двоично-десятичный и др. Отсюда возникает задача преобразования кодов.

Функциональные цифровые узлы комбинационного типа, переводящие коды из одного вида в другой, относятся к преобразователям кодов. Рассмотрим три типа подобных устройств: шифраторы, преобразователь кода для семисегментного индикатора и дешифраторы.

Шифраторы

Шифраторы, называемые также кодерами (в англоязычной литературе – encoder, coder, cipherer), могут осуществить преобразование десятичных чисел (позиционный код) в двоичную систему счисления В УГО шифраторов на рабочем поле делают пометку ENC от англ ENCODER.

Построим схему шифратора, который при нажиме на клавишу с соответствующим номером (позицией, разрядом) дает на своем выходе двоичный код этого числа. Пусть входной позиционный код имеет восемь разрядов X_K, где к=0...7 – номер разряда Тогда выходной двоичный код Y_K должен иметь три разряда. Y_0, Y_1, Y_2 Соответствие между X_K и Y_K выражается следующей ТИ

Таблица 1

№	Позиционный код								Двоичный код		
	X_7	X_6	X_6	X_4	X_3	X_2	X_1	X_0	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0	0	0	1
2	0	0	0	0	0	1	0	0	0	1	0
3	0	0	0	0	1	0	0	0	0	1	1
4	0	0	0	1	0	0	0	0	1	0	0
5	0	0	1	0	0	0	0	0	1	0	1
6	0	1	0	0	0	0	0	0	1	1	0
7	1	0	0	0	0	0	0	0	1	1	1

Из анализа этой таблицы (см. табл. 1), напоминающей турнирные таблицы спортивных чемпионатов, видно, что каждый разряд выхода в двоичном коде может быть получен четырехвходовым БЛЭ *ИЛИ* (OR), причем на его входы должны подаваться сигналы в соответствии с построчной связью единиц в X_K и Y_K. Например, для самого младшего значащего разряда в двоичном коде Y_0, соответствующего 2^0, на входы БЛЭ надо подать сигналы с линий X_1, X_3, X_5 и X_7. По аналогии нетрудно определить также связи Y_1 и Y_2.

Перейдем к построению схемы шифратора.

В программе **EWB** выберем восемь ключей, источник +5 V, заземление, три четырехвходовых БЛЭ OR и три логических ин-

дикатора LED. Проведя необходимые соединения и редактирование, получим схему шифратора с восемью входами и тремя выходами (см. рис. 86).

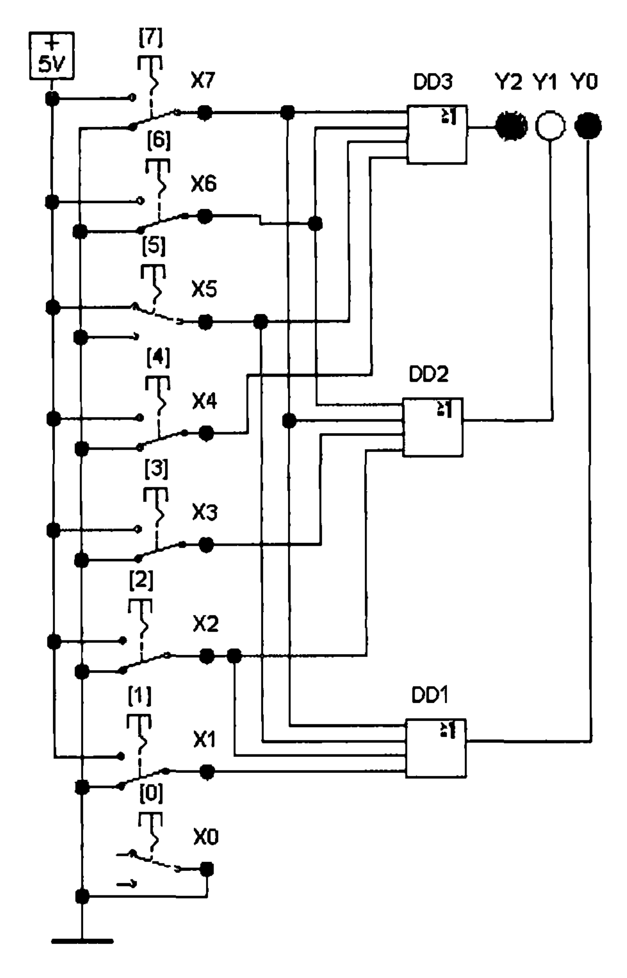


Рис. 86. Логическая структура шифратора 8×3 (EWB)

Подобный цифровой узел называется шифратором 8×3. Нажимая на одну из клавиш, видим, как на выходном индикаторе из трех ламп, расположенных упорядоченно по разрядам двоичных чисел, можно прочитать соответствующее двоичное число (не

горящий индикатор – 0, горящий – 1). Так, поскольку на рис. 86 нажат ключ 5, то картина на индикаторах такова: Y_3=1, Y_2=0, Y_0=1, что соответствует двоичному числу 101 Нулевой вывод (X_0) здесь является вырожденным.

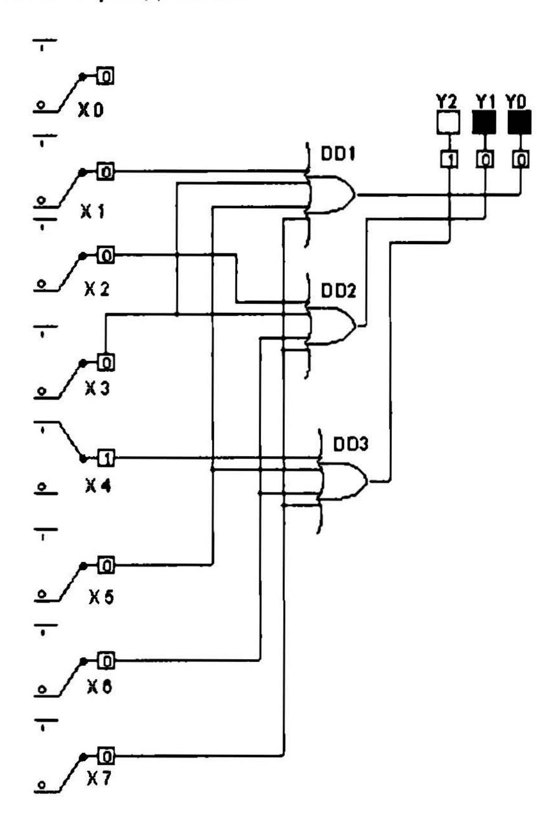


Рис. 87 Логическая структура шифратора 8×3 (**МС**)

В программе **МС** строим аналогичный шифратор (см рис 87). Здесь нажат ключ 4, и картина на индикаторах такова: Y_3=1, Y_2=0, Y_0=0, что соответствует двоичному числу 100

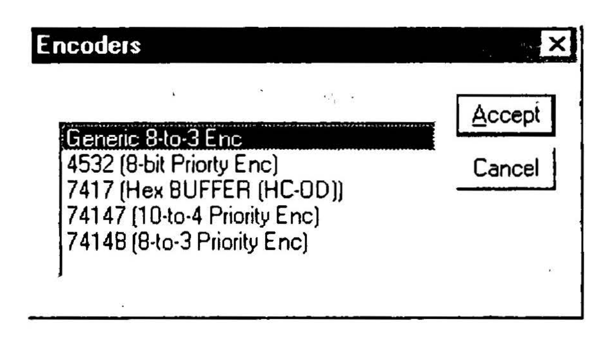
Одновременное нажатие более чем на один ключ в этих шифраторах не допускается (не учитывая нулевого ключа, являющегося холостым).

Реализация шифраторов на микросхемах

Рассматривая схемы шифраторов, составленные нами из отдельных БЛЭ, и обратившись к библиотеке компонентов программ, можно подобрать цифровые микросхемы, содержащие необходимые элементы, и провести из них сборку виртуальных моделей. Однако это не даст нам ничего нового и вряд ли целесообразно, так как в этих же библиотеках содержатся готовые модели выпускаемых микросхем. Последние обладают также дополнительными возможностями по сравнению с рассмотренными нами упрощенными моделями. В микросхемном исполнении имеются так называемые *приоритетные* шифраторы, в которых при нажатии на несколько клавиш на выход будет выведен номер большей из них Для этого микросхема внутри дополняется логикой выбора приоритета.

В программе **EWB** выбор необходимой микросхемы выполняется из панели Digital пиктограммой ENC, открывающей меню библиотечных шифраторов (см рис 88,а).

Примером приоритетного шифратора 10×4 может служить микросхема ТТЛ 74147, которую можно использовать в клавиатуре На рис. 88,б показана ТИ этой микросхемы, взятая из раздела предметной помощи программы **EWB**. Нулевая линия (вход) отсутствует как в таблице, так и в последующей схеме: ее роль выполняет информационное заземление. Несмотря на это счет здесь (как, впрочем, и везде в цифровой электронике) ведется, начиная с 0 Поэтому, несмотря на обозначенные девять входов шифратор называется 10×4 Особенностью данной микросхемы в том, что для нее активным является низкий уровень входного сигнала. Поэтому если на входы не подается напряжение низкого уровня, то на всех выходах устанавливается напряжение высокого уровня, соответствующее десятичному числу 0. Когда же на один из входов подается 0, то на выходе формируется двоичный код номера входа. При подаче нуля на несколько входов приоритет отдается входу с высшим номером, а остальные входы игнорируются. Это обстоятельство подчеркнуто в ТИ тем, что в ее левой половине выше нулевой диагонали проставлены косые кресты (безразлично 0 или 1). Так, например, подача 0 на вход 9 приводит к игнорированию всех остальных входов, а выход примет значение 0110, что соответствует в отрицательной логике десятичной цифре 9. Если провести поразрядную инверсию вы-



а)

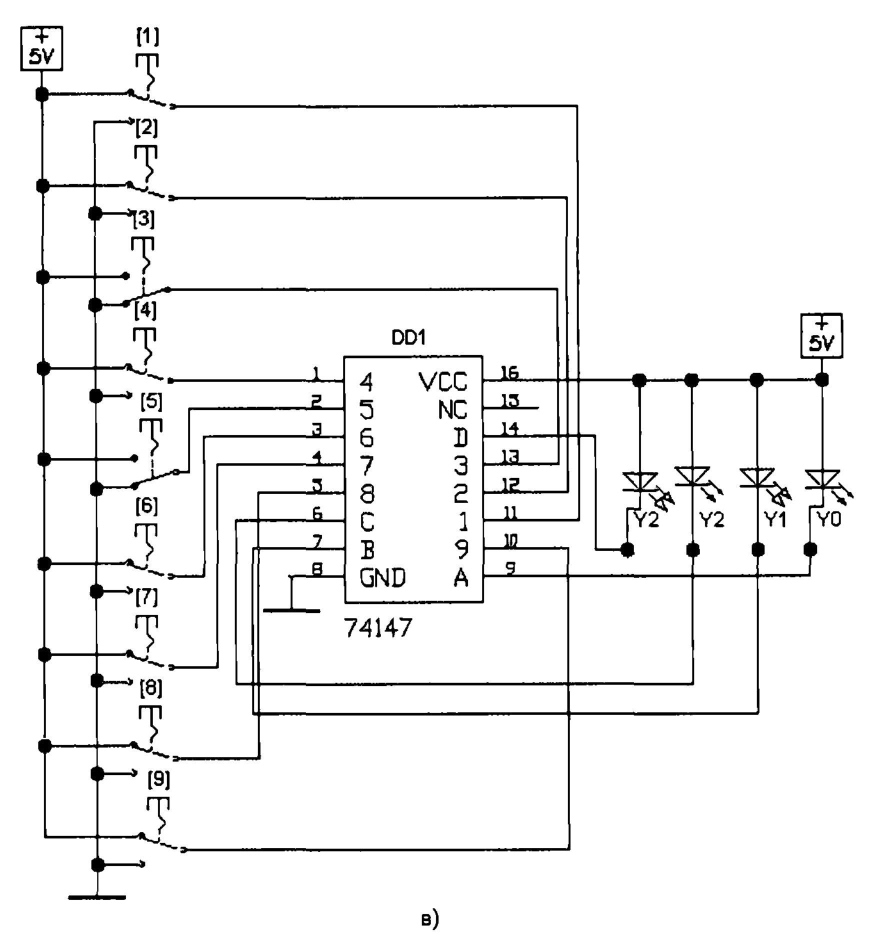
74147 (10-to-4 Priority Enc)

10-Line to 4-Line Priority Encoder truth table:

Inputs									Outputs			
1	2	3	4	5	6	7	8	9	D	C	B	A
1	1	1	1	1	1	1	1	1	1	1	1	1
X	X	X	X	X	X	X	X	0	0	1	1	0
X	X	X	X	X	X	X	0	1	0	1	1	1
X	X	X	X	X	X	0	1	1	1	0	0	0
X	X	X	X	X	0	1	1	1	1	0	0	1
X	X	X	X	0	1	1	1	1	1	0	1	0
X	X	X	0	1	1	1	1	1	1	0	1	1
X	X	0	1	1	1	1	1	1	1	1	0	0
X	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0

б)

Рис. 88 (начало)



в)

Рис. 88 (окончание) Приоритетный шифратор 10×4 на микросхеме 74147 (**EWB**)

ходного сигнала (с учетом того, что 0`=1 и 1`=0), то получим 0`1`1`0`=$(1001)_2$, т е 9_{10}. Сборка шифратора на данной микросхеме показана на рис 88,б Как и ранее к выводу VCC подключаем питание, GND – заземляем, NC (No Connection – не соединен) – это холостой вывод, 1–9 – входы; A,B,C,D – выходы Поскольку выходы являются инверсными, то для зажигания выходных светоизлучающих диодов их аноды подключены к плюсу источника постоянного напряжения, а катоды к соответствующим выводам

микросхемы. Появление 0 на выходе, таким образом, соответствует заземлению СИД, и он загорается. На рис. 88,в показана работа приоритетного шифратора, когда нажаты, т.е. поданы нули на клавиши 3 и 5. Показание индикатора в двоичном коде 0101, т е. 5_{10} Число 3 проигнорировано, иначе во втором разряде была бы 1, а не 0.

Тот же клавиатурный шифратор показан в программе **MC** (см рис 89). В отличие от программы **EWB** здесь разводка выводов не соответствует той, что имеется на реальной микросхеме, зато в явном виде показано, что входы и выходы инверсны На рис 89 нажаты клавиши 5 и 9. Выход имеет вид 0110, т.е 9_{10}, если учесть инверсию (1001).

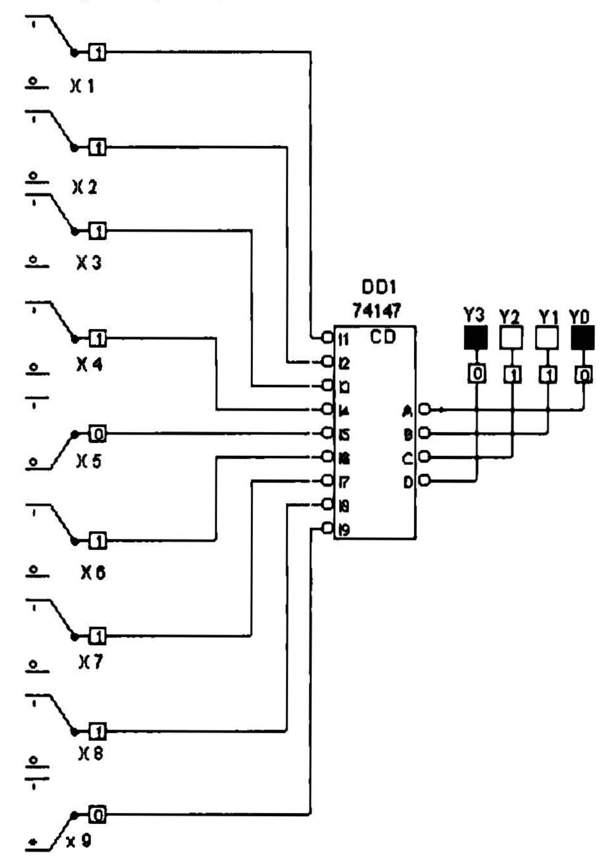


Рис. 89. Приоритетный шифратор 10×4 на микросхеме 74147 (**MC**)

При использовании шифраторов в ЦУ часто бывает необходимо их каскадирование Примером подобного шифратора является приоритетный шифратор 8×3 на микросхеме ТТЛ 74148 В схему вводят один дополнительный вход EI (Enable Input) – разрешение входа и два дополнительных выхода GS (Group Signal) – групповой сигнал и EO (Enable Out) – разрешение выхода Согласно ТИ, показанной на рис 90,а, на выходе GS формируется напряжение низкого уровня, если такое напряжение подается на один из входов На выходе EO формируется напряжение низкого уровня, когда на все входы подается напряжение высокого уровня На вход EI при нормальной работе подается напряжение низкого уровня Примеры включения приоритетных шифраторов 8×3 микросхемы ТТЛ 74148 показаны на рис. 90,б и 91.

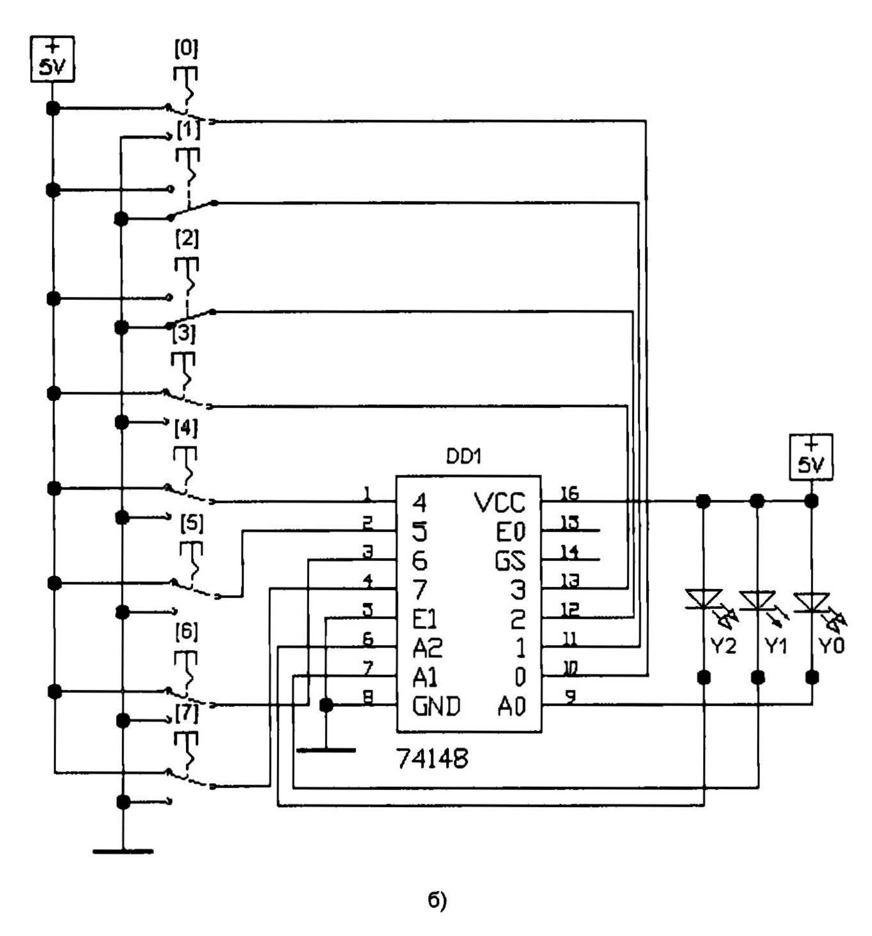
74148 (8-to-3 Priority Enc)

8-Line to 3-Line Priority Encoder truth table:

EI	Inputs 0	1	2	3	4	5	6	7	Outputs A2	A1	A0	GS	EO
1	X	X	X	X	X	X	X	X	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	0
0	X	X	X	X	X	X	X	0	0	0	0	0	1
0	X	X	X	X	X	X	0	1	0	0	1	0	1
0	X	X	X	X	X	0	1	1	0	1	0	0	1
0	X	X	X	X	0	1	1	1	0	1	1	0	1
0	X	X	X	0	1	1	1	1	1	0	0	0	1
0	X	X	0	1	1	1	1	1	1	0	1	0	1
0	X	0	1	1	1	1	1	1	1	1	0	0	1
0	0	1	1	1	1	1	1	1	1	1	1	0	1

а)

Рис 90 (начало)



б)

Рис 90 (окончание). Приоритетный шифратор 8×3 на микросхеме 74148 (**EWB**)

Из приведенных рисунков виден смысл функции приоритета: активный низкий уровень здесь подан одновременно на входы 1 и 2, а на выходе в двоичном коде зажигается индикатор Y1 (см рис 90,а), при этом сигнал с входа 1 игнорируется (в программе **MC** на рис 91 индикация дана в инверсном коде)

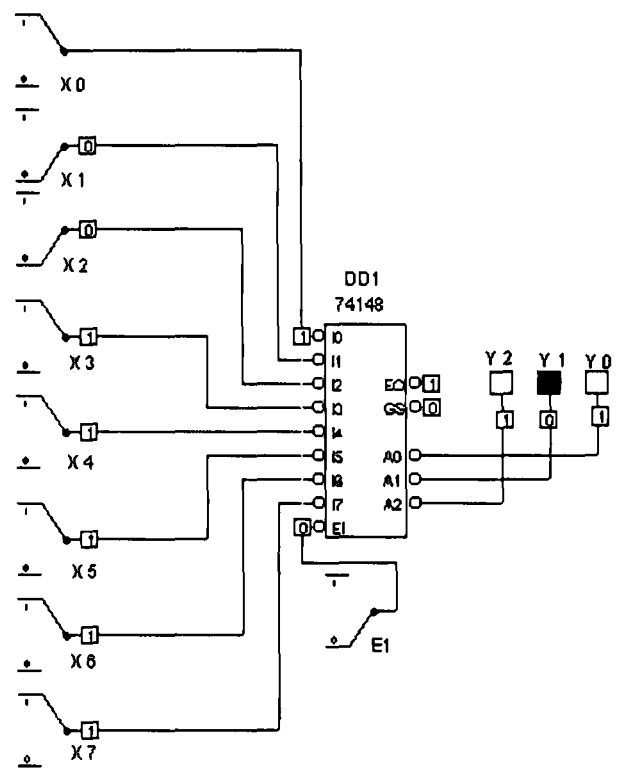


Рис. 91. Приоритетный шифратор 8×3 на микросхеме 74148 (**МС**)

Дешифраторы

Дешифратор, или декодер (англ. – Decoder, сокращенно – DC), выполняет обратную по отношению к шифрованию операцию, т.е. преобразует двоичный код в десятичный. Входы дешифратора служат для подачи двоичных чисел, а выходы последовательно нумеруются десятичными числами. При подаче на входы двоичного числа выходной сигнал появляется на выходе, который имеет номер соответствующего десятичного числа.

Существует два типа дешифраторов: логические дешифраторы и дисплейные дешифраторы/формирователи. Логические дешифраторы представляют собой схемы средней интеграции (микросхемы, имеющие в своем составе до 100 ЛЭ), управляемые адресом Они выбирают и приводят в активное состояние

конкретный выход, определяемый адресом. Дешифраторы применяются в структурах выборки адреса запоминающих устройств, разуплотнения маршрутизации данных и т.п

Таблица преобразования для дешифратора получается из предыдущей (шифратора), если в ней поменять местами функцию и аргумент Тогда таблица декодирования примет следующий вид.

Таблица 2

№	Двоичный код			Позиционный код							
	X_2	X_1	X_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
2	0	1	0	0	0	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0	0	0
4	1	0	0	0	0	0	1	0	0	0	0
5	1	0	1	0	0	1	0	0	0	0	0
6	1	1	0	0	1	0	0	0	0	0	0
7	1	1	1	1	0	0	0	0	0	0	0

Как следует из данной таблицы, соответствующий выход позиционного кода для 1: $(X_2)`(X_1)`X_0$, для 2 $(X_2)`X_1(X_0)`$, для 7: $X_2X_1X_0$ Примем X_0=A, X_1=B и X_2=C и построим соответствующий задатчик двоичных чисел на ключах, снабдив его индикацией каждого двоичного разряда (см. рис. 92).

Возьмем семь БЛЭ DD0–DD7 (номер элемента соответствует десятичной цифре позиционного кода), три инвертора DD8–DD10 и проведем соединения в соответствии с логикой преобразования (см. рис. 92). Для проверки работоспособности полученной схемы на выходах расположены индикаторы.

Входные и выходные сигналы на рис. 92 интерпретируются следующим образом. При C=1, B=1 и A=0, т.е. при сигнале на входе $(110)_2$, сигнал на выходе соответствует 6_{10}, что и соответствует операции декодирования.

Далее полученный дешифратор оформим в виде субблока DC1 (рис. 93).

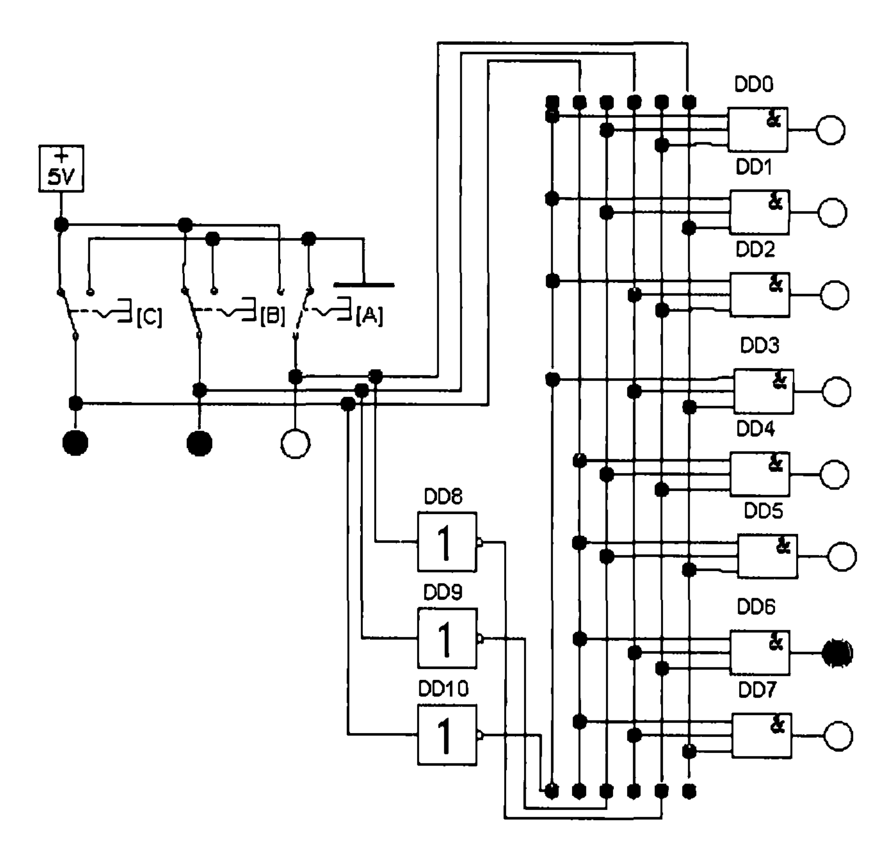
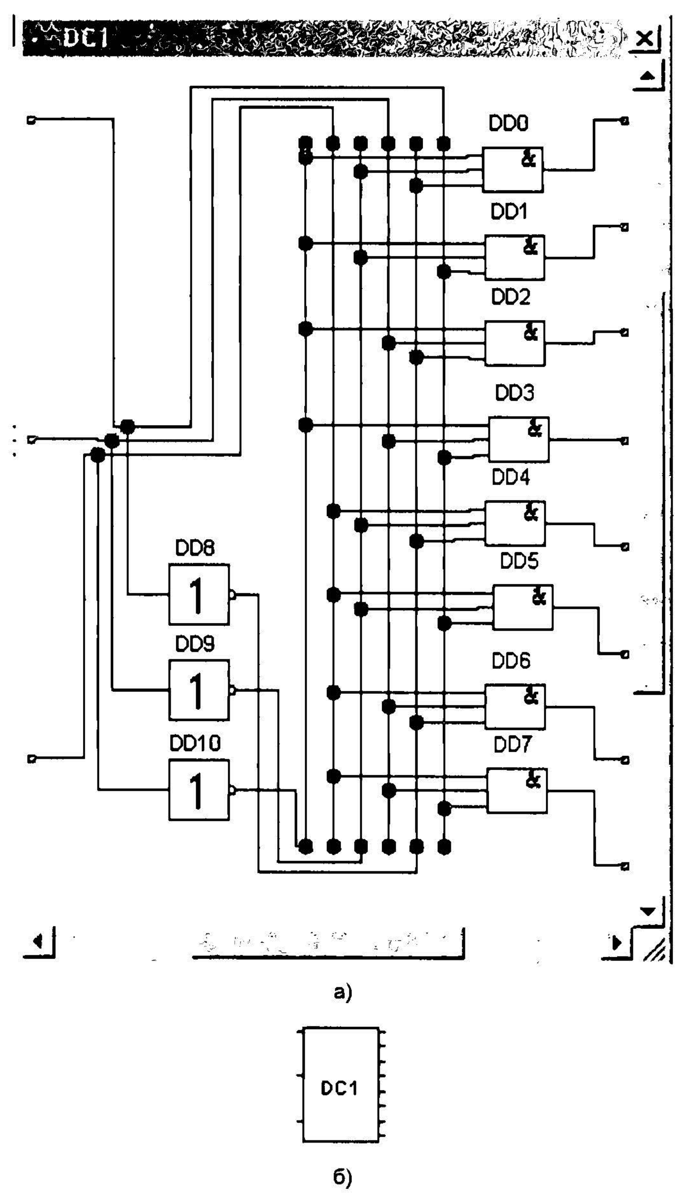


Рис. 92 Логическая структура дешифратора (**EWB**)

Аналогичная схема дешифратора может быть создана и в программе **МС** (см рис 94). Задавая здесь двоичные числа ключами X_0, X_1 и X_2 и запуская моделирование, можно пронаблюдать соответствие двоичных чисел на входе и номера «к» выхода $Y_к$, в котором возникает сигнал высокого уровня



а)

б)

Рис. 93. Субблок дешифратора (**EWB**)

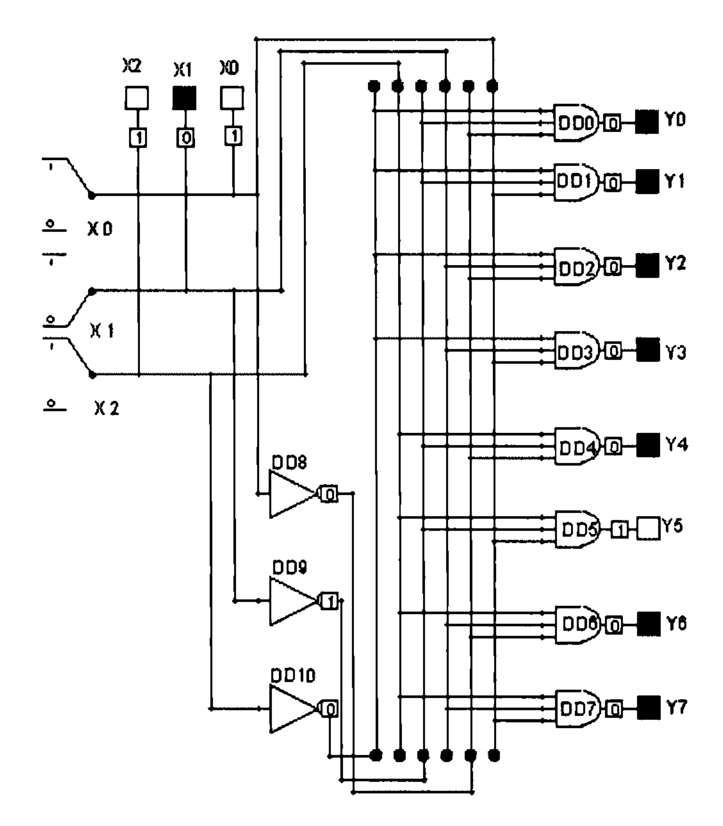


Рис. 94. Логическая структура дешифратора (**МС**)

Преобразователь кода для семисегментного индикатора

Дешифраторы и дисплейные дешифраторы/формирователи формируют цифровые коды для 7-сегментного индикатора, и затем обеспечивают передачу кода на формирователь или непосредственно на дисплей. В семисегментном индикаторе десятичных цифр каждый сегмент Y_K, где к = 1 7, представляет собой отдельный светоизлучающий элемент (используется также буквенная идентификация сегментов, соответственно от *a* до *g*) Светящееся изображение цифр или знаков получается при подаче напряжения на определенные сегменты.

Работу индикатора можно посмотреть в программе **МС**. Для этого надо выбрать индикатор командами: Component>Animation>

Seven Segment. В результате на рабочем поле появится семисегментный индикатор (см. рис. 95,а, на котором дополнительно показаны буквенные метки сегментов). Подключив семь цифровых ключей на входы индикатора и задавая на них высокий уровень, можно получить на индикаторе различные цифры и буквы

Совет **(MC)** Не забывайте, что после каждой смены положений цифровых ключей необходимо заново подать команду на проведение моделирования (Run).

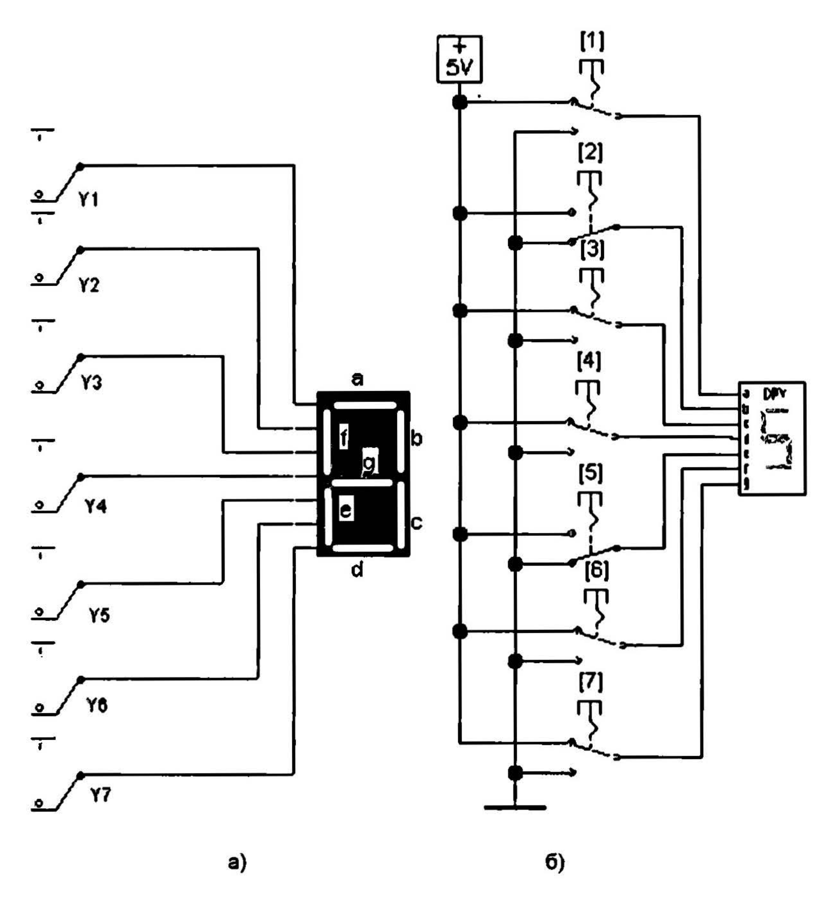


Рис. 95. Семисегментный индикатор: а – **MC**; б – **EWB**

Аналогичную схему соберем в программе **EWB** (рис. 95,б), в которой семисегментный индикатор (по-английски – Seven-Segment Display), выбирается в панели компонентов Indicators - по его пиктограмме .

Как следует из вышеописанного, преобразователь входного двоичного кода X_K (X_3, X_2, X_1, X_0) в выходной код управления семисегментным индикатором Y_K должен удовлетворять ТИ, представленной ниже для положительной логики

Таблица 3

№	X_K				Y_K						
	X_3	X_2	X_1	X_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
					a	*b*	*c*	*d*	*e*	*f*	*g*
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	0	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	0	0	1	1
A	1	0	1	0	1	1	1	0	1	1	1
B	1	0	1	1	0	0	1	1	1	1	1
C	1	1	0	0	1	0	0	1	1	1	0
D	1	1	0	1	0	1	1	1	1	0	1
E	1	1	1	0	1	0	0	1	1	1	1
F	1	1	1	1	1	0	0	0	1	1	1

Здесь числа от 10 до 15 представлены начальными заглавными буквами латинского алфавита соответственно от A до F, так как это принято в шестнадцатеричном коде. Кодировка отдельных чисел в преобразователях кода конкретных микросхем может быть и иной. Если индикатор работает при активном низком уровне (в отрицательной логике), то правая часть ТИ (Y_K) должна быть инвертирована.

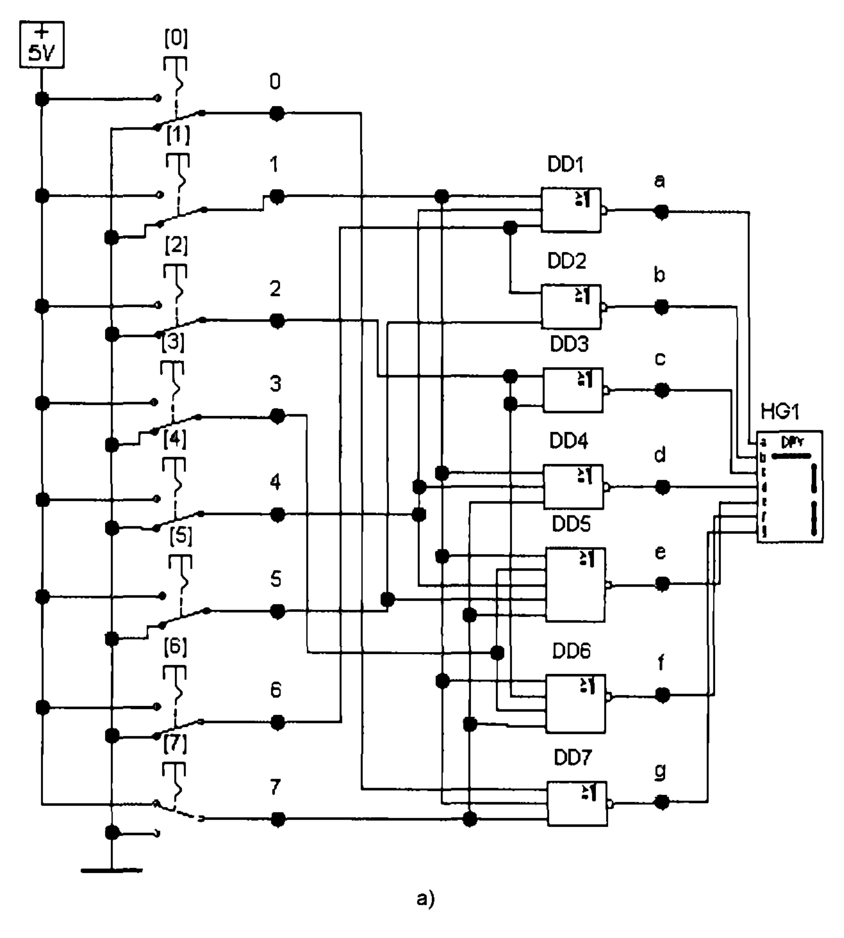
Задача преобразования двоичного кода в код управления семисегментным индикатором может быть решена различными способами. Для большей наглядности результата ограничимся тремя разрядами двоичного кода и разобьем исходную задачу на две. вначале преобразуем двоичный код в позиционный, а затем позиционный – в код управления семисегментным индикатором

Первая часть задачи была только что решена: ее выполняет дешифратор. При решении второй части задачи обратим внимание на то, что ТИ имеет в выходных переменных гораздо меньше нулей, чем единиц В этом случае будет рациональнее составлять логическую функцию воспользовавшись операцией дизъюнкции Поскольку схемный семисегментный индикатор зажигается при высоком потенциале, то полученную функцию надо инвертировать В результате мы приходим к выводу, что надо использовать БЛЭ Пирса, т.е. NOR. Таких элементов нам потребуется столько, сколько сегментов на индикаторе, т е. семь Обозначим их малыми латинскими буквами от *a* до *g*, которыми обозначены сегменты Количество входов каждого элемента равно числу нулей в соответствующем выходном столбце (см. табл. 3) Для а – 3, для *b* – 2 и т.д. Наибольшее число входов – 6 – имеет элемент *e*. На входе с вообще один 0, значит там можно обойтись инвертором, но, имея в виду дальнейшую микросхемную реализацию, этот инвертор образуем из двухвходового ЛЭ NOR. Используя ранее выполненный задатчик двоичного кода и дешифратор, проведем сборку схемы, руководствуясь ТИ. На выходе устройства подключим семисегментный индикатор. В результате получим схему, показанную на рис 96,а.

Часть этой схемы, содержащую преобразователь позиционного кода в код семисегментного индикатора, также оформим в виде отдельного субблока, присвоив ему метку DC2 (рис. 96,б). Построим схему с дешифраторами DC1 и DC2 (рис. 97, а). Затем объединим два субблока DC1 и DC2 в один блок DC3 (см. рис. 97,б,в) – это и будет дешифратор-драйвер для семисегментного индикатора (рис. 97,г).

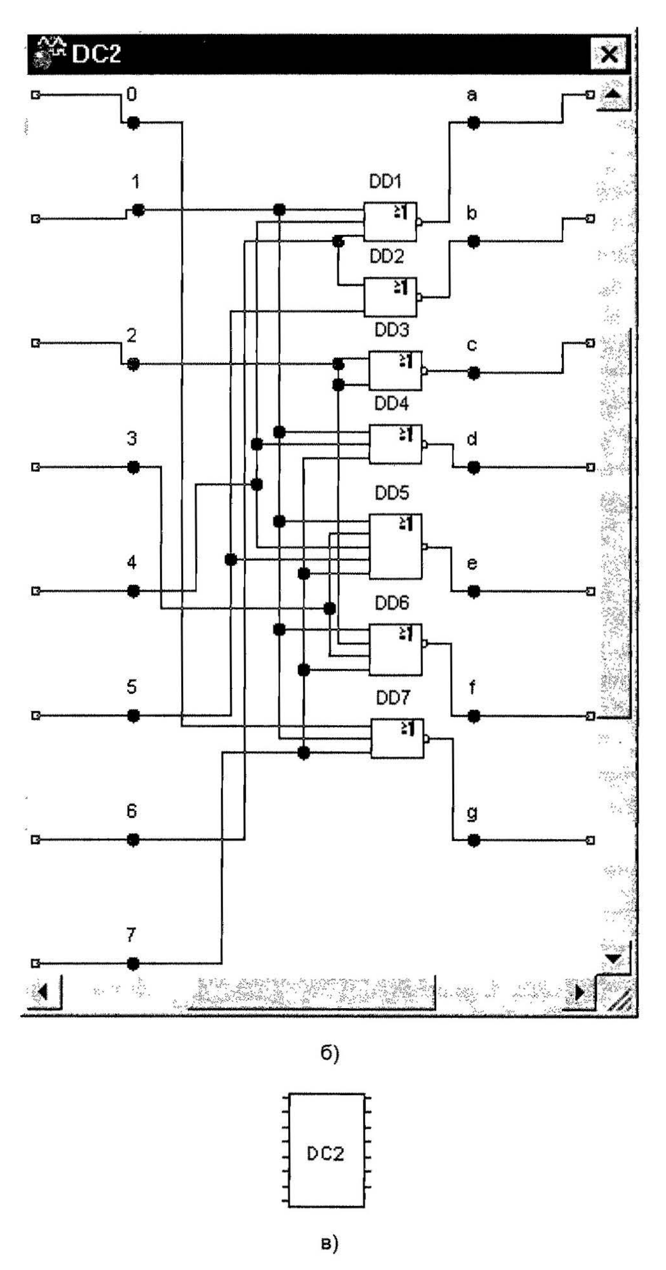
В программе **EWB** подобное устройство имеется в готовом виде, но оно уже как бы «зашито» внутрь виртуального семисегментного индикатора. В этом случае последний имеет четыре входа, а не семь. Дополнительный четвертый вход (по сравнению с тремя рассмотренными) дает еще один разряд для входного двоичного кода. В результате на выходе можно получить максимальное число не 7, а – 15 (F). Данное устройство, названное в программе Decoded Seven-Segment Display (декодирующий се-

мисегментный индикатор), находится в панели индикаторов и выбирается по его пиктограмме . Дополнив задатчик двоичного кода еще одним разрядом (D) и подключив его на вход индикатора, убеждаемся в работоспособности схемы, показанной на рис. 98.



a)

Рис 96 (начало)



б)

в)

Рис. 96 (окончание). Дешифратор позиционного кода в код семисегментного

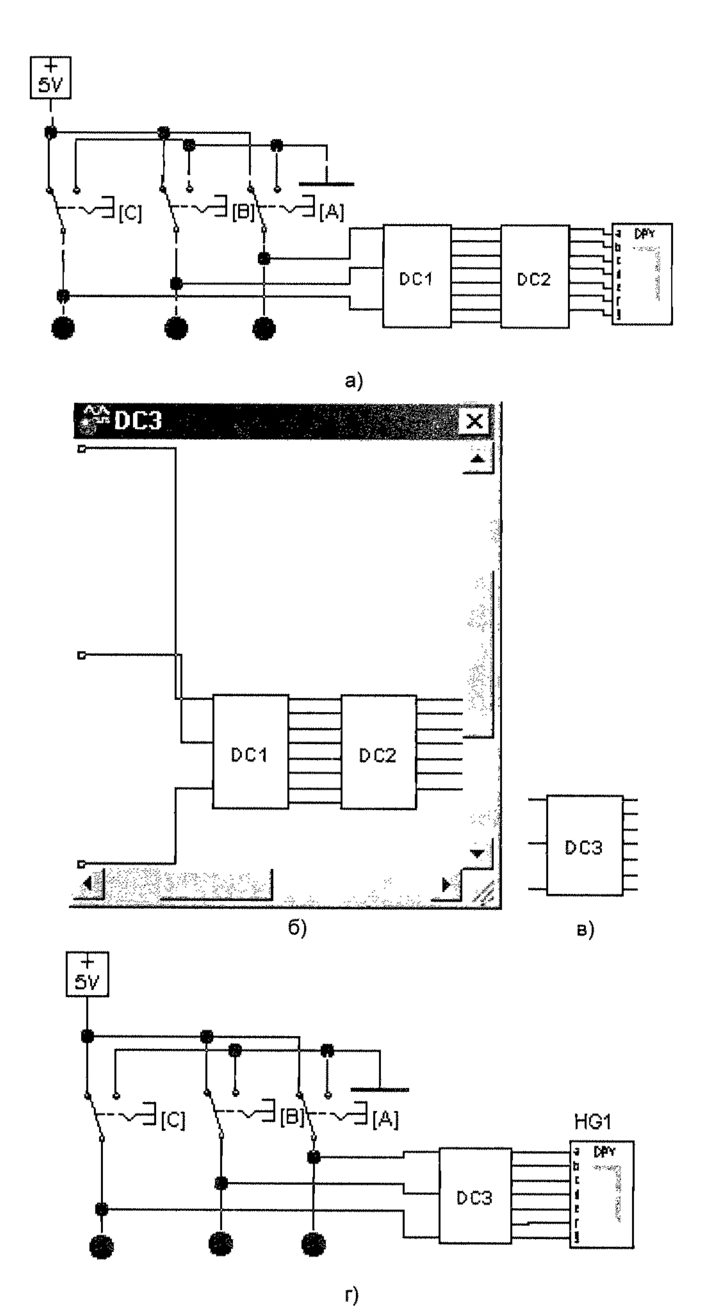


Рис. 97. Дешифратор двоичного кода в код семисегментного индикатора

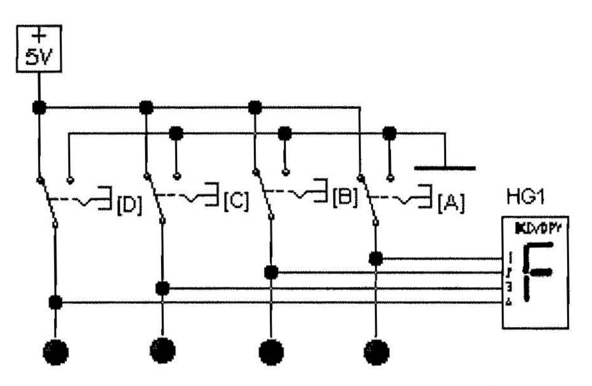


Рис 98 Декодирующий семисегментный индикатор (**EWB**)

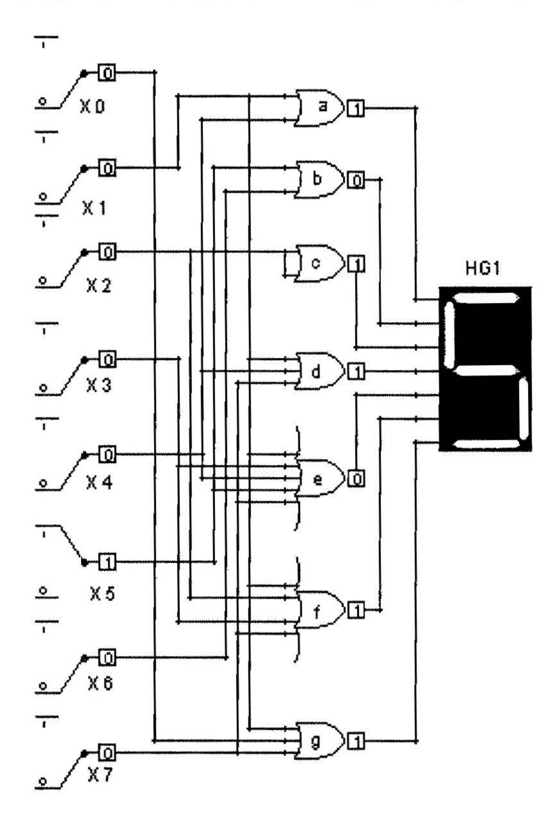


Рис. 99. Дешифратор позиционного кода в код семисегментного индикатора (**MC**)

В программе **МС** также соберем схему выходной части драйвера индикатора (см. рис. 99) и, проверив ее работоспособность, составим полную схему устройства. Для этого скопируем из соответствующих файлов в одно рабочее окно схемы, показанные на рис. 94 и 99. Затем устраним лишние элементы и проведем соединения выходов первого дешифратора с входами второго и окончательно получим дешифратор-драйвер для семисегментного индикатора (рис. 100).

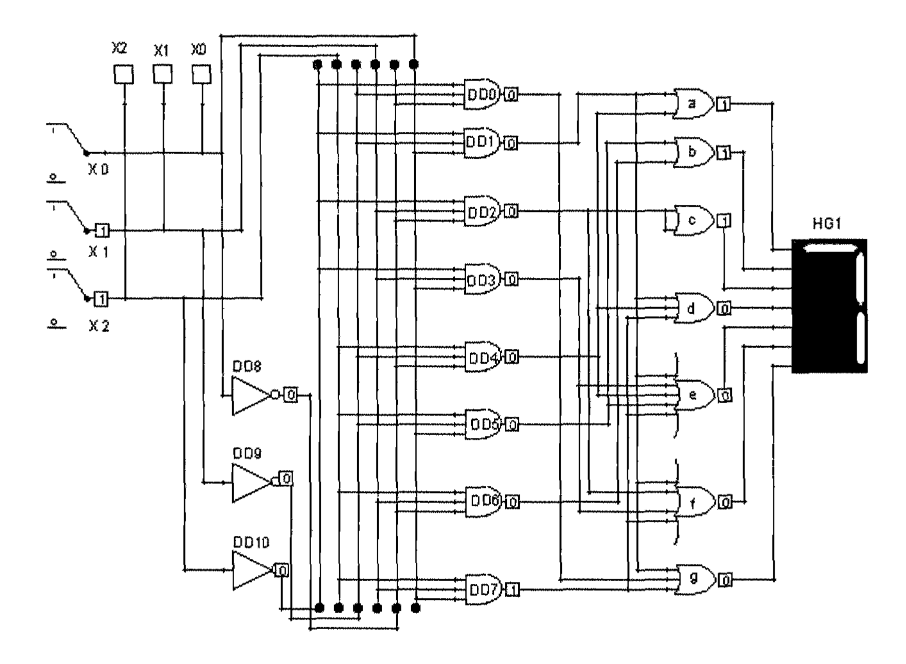


Рис. 100. Логическая структура дешифратора двоичного кода в код семисегментного индикатора (**МС**)

Свертка этой схемы в программе **МС** так, чтобы получился аналог схемы, показанной на рис. 97 в программе **EWB**, в принципе возможна, но требует привлечения специальных приемов программирования

В библиотеках обеих программ содержатся модели компонентов стандартных микросхем, служащих дешифраторами. Примером могут служить дешифраторы для управления семисегментным индикатором на микросхемах ТТЛ 7447 и 7448. На

рис. 101,а показана ТИ микросхемы 7447, имеющей на выходе открытый коллектор и включение микросхемы 7448, имеющей на выходах высокий потенциал (см. рис. 101,б,г). Данный дешифратор преобразует двоично-десятичный код, соответственно на английском языке – Binary-Decimal Code (BCD), подаваемый на входы A, B, C, D, в код управления 7-сегментным индикатором.

Двоично-десятичный код представляет собой упорядоченный по разрядам набор двоичных чисел, в котором разрядам приписаны следующие «веса» в порядке уменьшения старшинства. D – 8, C – 4, B – 2, A – 1 Поэтому данный код называют также кодом 8-4-2-1 Фактически в этом коде записаны десятичные числа от 0 до 15 во входных переменных ТИ (см. рис. 101,а).

7447 (BCD-to-Seven-Segment Dec)

No.	$\overline{LT}$	$\overline{RBI}$	D C B A	$\overline{BI}/\overline{RBO}$	a b c d e f g		
0	1	1	0 0 0 0	1	1 1 1 1 1 1 0		
1	1	X	0 0 0 1	1	0 1 1 0 0 0 0		
2	1	X	0 0 1 0	1	1 1 0 1 1 0 1		
3	1	X	0 0 1 1	1	1 1 1 1 0 0 1		
4	1	X	0 1 0 0	1	0 1 1 0 0 1 1		
5	1	X	0 1 0 1	1	1 0 1 1 0 1 1		
6	1	X	0 1 1 0	1	0 0 1 1 1 1 0		
7	1	X	0 1 1 1	1	1 1 1 0 0 0 0		
8	1	X	1 0 0 0	1	1 1 1 1 1 1 1		
9	1	X	1 0 0 1	1	1 1 1 0 0 1 1		
10	1	X	1 0 1 0	1	0 0 0 1 1 0 1	<	I
11	1	X	1 0 1 1	1	0 0 1 1 0 0 1	<	N
						<	V
12	1	X	1 1 0 0	1	0 1 0 0 0 1 1	<	A
13	1	X	1 1 0 1	1	1 0 0 1 0 1 1	<	L
14	1	X	1 1 1 0	1	0 0 0 1 1 1 1	<	I
15	1	X	1 1 1 1	1	0 0 0 0 0 0 0	<	D
BI	X	X	X X X X	0	0 0 0 0 0 0 0		
RBI	1	0	0 0 0 0	0	0 0 0 0 0 0 0		
LT	0	X	X X X X	1	1 1 1 1 1 1 1		

а)

Рис. 101 (начало)

б)

в)

г)

Рис. 101 (окончание). Дешифратор двоичного кода в код семисегментного индикатора на микросхеме 7448.

Рассматриваемый дешифратор в отличие от предыдущих имеет дополнительные инверсные входы: (LT) – Lamp Test – контроль индикатора; (BI/RBO) – Blanking Input/ Ripple Blanking Output – вход гашения/выходной сигнал схемы последовательного гашения; (RBI) – Ripple Blanking Input – входной сигнал схемы последовательного гашения. На все эти выводы микросхемы при нормальной работе надо подать высокий уровень напряжения. Активный уровень на выходе – высокий. Графика изображения отдельных десятичных цифр и чисел, выводимых на индикатор, мягко говоря, отличается от традиционной арабско-индийской (см. ряд экранных снимков с индикатора на рис 101,в) Так, у цифр 6 и 9 отсутствуют соответственно верхняя и нижняя «перекладины». Начиная с числа 10, комментировать вообще сложно Когда такая абракадабра появляется на экране, начинаешь искать ошибки в схеме. Можно, конечно, попытаться разгадать этот ребус Число 10 в представленном виде – это комбинация единицы (левая вертикальная «стойка») и «ущербного» нуля (у него нет правой вертикальной «стойки»), в 11 то же самое, но 1 есть справа, а не слева; дальнейшие числа 12...14 это как бы то, что надо добавить к восьми, чтобы получилось соответствующее число, причем его надо догадаться взять из части предыдущих изображений, т.е. 12=8+4, и у четверки надо взять только верхнюю часть и т д Наконец, 15 – пустой дисплей (в отличие от нуля, изображенного в полную силу). Вот и догадывайся, что это показание, а не выключенный экран. Остается только поблагодарить неизвестных нам изобретателей за такие «подарки», а то ведь могли бы еще использовать египетскую или китайскую нумерацию Теперь во всех справочниках по микросхемам печатают эти изображения, а в ТИ строки с 10 по 15 выделяют серым цветом (помните, товарищи!) или, как остроумно поступили разработчики программы **EWB**, поместив справа от этих строк глубокомысленную вертикальную помету: INVALID (см. рис 101, а) Впрочем, существуют и другие способы кодировки, когда 10_{16}на индикаторе высвечивается как **A**, 11_{16} изображается строчной латинской буквой **b** (тогда цифра 6 изображается полностью), 12_{16} – либо заглавной **C**, либо строчной **c**, 13_{16} – **D** или **d**, 14_{16}=**E**, а 15_{16}=**F**. Здесь работают простые зрительные ассоциации, и это представляется весьма рациональным (Кстати, исторически, именно буквенный способ использовался еще в древнеармянской нумерации. Подобная «кодировка» и по сей день применяется для обозначения глав книг, формул и т. п.)

В программе **МС** работа такой же микросхемы показана на рис. 101,г. По существу, единственное различие – это добавление в обозначениях входов слова BAR (стойка, полоска; возможно, контактная площадка): LTBAR, RBIBAR. Иногда этот термин пишут, усекая до одной буквы, например, PREB (PresetBar) – вход предустановки

2.2. Мультиплексоры и демультиплексоры

Покупайте два в одном флаконе! –
это выгодно и удобно.
Реклама

Мультиплексоры

В практике использования различных радиоэлектронных устройств часто возникает необходимость выбора (селекции) сигналов, поступающих на данный вход (линию) от одного из нескольких других. Так, в старых радиоприемниках и телевизорах подобную функцию (выбора диапазона или канала) осуществляли при помощи многопозиционного механического переключателя. В телефонии эту же роль выполняли вначале барышни-телефонистки, а затем – специальное электромеханическое устройство – шаговый искатель. Малая надежность подобных устройств очевидна. Кроме того, их габариты таковы, что трудно даже представить их совместимость с микроэлектроникой: на наших столах должны были бы оказаться груды железа. Фактически двухпозиционный ключ (Switch), который мы до сих пор многократно использовали для выбора лог. 0 или лог. 1, на входе ЦУ является простейшим контактным мультиплексором (так сказать – «дуплексором»)

Мультиплексор – это управляемый переключатель цифровых сигналов, обеспечивающий передачу одного из многих входных сигналов, чей номер задан двоичным кодом, на единственный выход Слово «мультиплексор» (англ. – Multiplexer, сокращенно – MUX) родственно следующим латинским словам multum – много, multiplicatio – умножать (мультипликация – множество рисунков-кадров), multiplex – сложный. Поскольку на входные линии мультиплексора подаются двоичные данные, то его также называют селектором данных Операция переключения из множества линий данных на одну носит название мультиплексирования

Рассмотрим мультиплексор, имеющий два входа данных D_0 и D1, одну управляющую линию их адресации (т е. указания двоичного кода выбираемой линии) А и выход Y. Этот двухвходовый мультиплексор можно назвать 2×1. Состояния подобного мультиплексора описывается следующей таблицей:

A	Y
0	D_0
1	D_1

Видно, что $Y=A\ D_0+AD_1$, а логическая структура, реализующая мультиплексор, должна иметь два БЛЭ типа AND, один инвертор и один элемент OR Откуда нетрудно собрать соответствующую схему

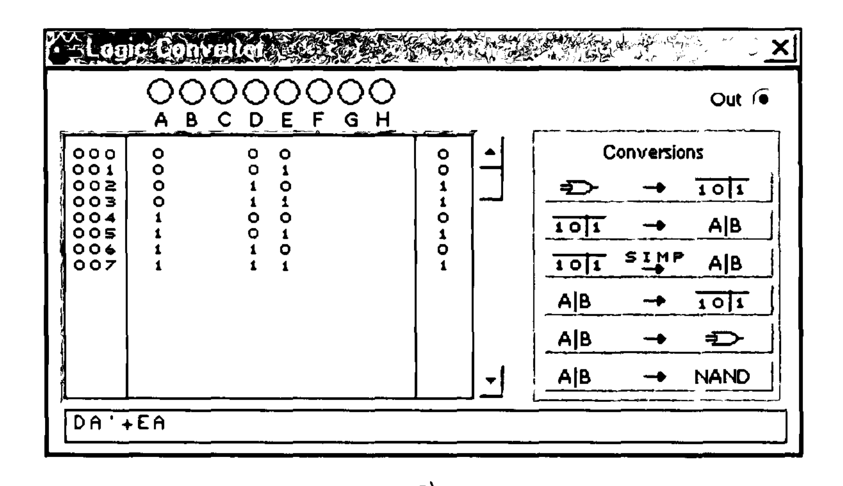
Можно также воспользоваться описанным ранее логическим конвертером программы **EWB**. Для этого, правда, придется преобразовать таблицу состояний в более подробную ТИ с учетом всех возможных вариантов комбинаций сигналов, приведя в соответствие обозначения входов с регламентированной в **EWB**. Пусть А – первый адресный вход (для последующих зарезервируем В и С), D – первая линия данных (соответствует D_0), выбираемая адресом А=0, Е – вторая линия данных, которая соответствует D_1, выбираемая адресом А=1 (для более сложных мультиплексоров зарезервируем обозначения входов данных F и G). Перепишем в этих переменных логическую функцию:

$Y=A\ D+AE$

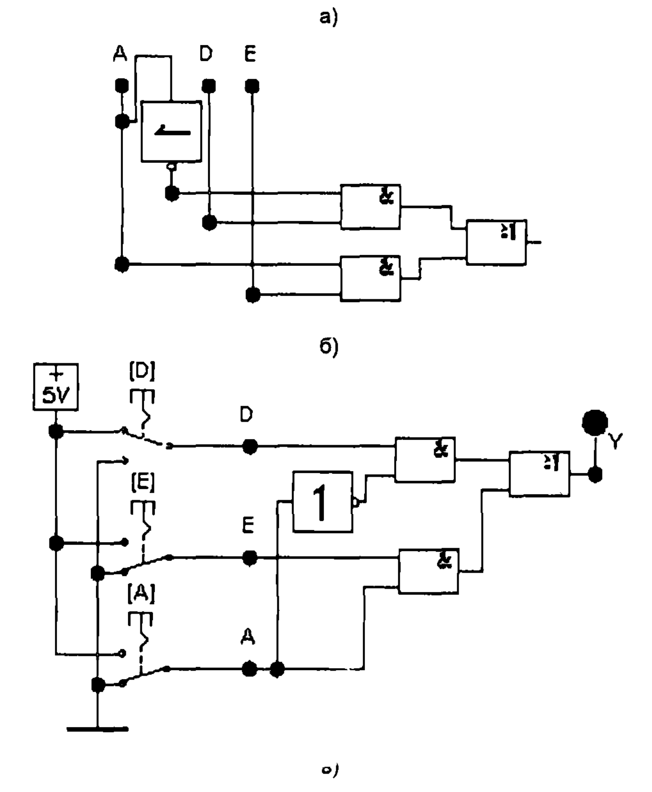
и введем ее в логический конвертер, нажав клавишу A|B → 1 0|1 (см. рис. 102,а).

Таким образом, ТИ мультиплексора 2×1 примет следующий вид:

№	A	D	E	Y
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1



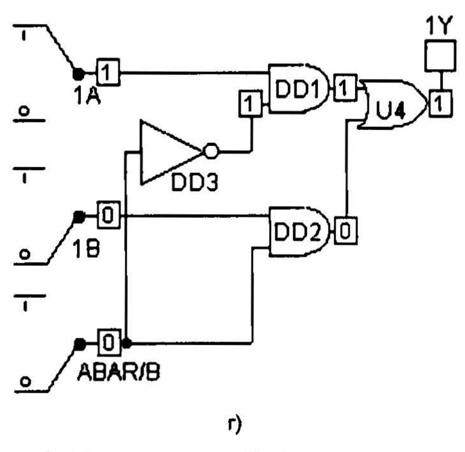
а)



б)

в)

Рис 102 (начало)



г)

Рис 102 (окончание) Мультиплексор 2×1

Нажав далее на клавишу , получим соответствующую схему (смотри рис 102,б, где приведен неотредактированный снимок схемы, выведенной на экран). Отредактировав эту схему и дополнив ее источниками сигналов и выходным индикатором, получим виртуальную модель мультиплексора 2×1 (рис 102,в). Включив моделирование и задав A=0, видим, что выход повторяет сигнал на D, а при A=1 – Y=E.

В программе **MC** мультиплексор 2×1 показан на рис 102,г Здесь адресный вход обозначен через ABAR/B и входы данных 1A и 1B, как это принято в обозначениях мультиплексора микросхемы ТТЛ 74157 в данной программе (см. ниже). Распределение сигналов (состояния цифровых узлов в виде 0 или 1 в прямоугольных рамочках около соответствующих узлов) видно из рис 102,г

Реализацию мультиплексора 2×1 в виде микросхем рассмотрим на примере ТТЛ 74157, которая представляет собой четыре подобных мультиплексора, собранных в одном корпусе. Выбрав эту микросхему в программе **EWB** из библиотеки, открываемой пиктограммой , соберем на первом мультиплексоре (1A, 1B, 1Y) соответствующую схему (рис. 103,а)

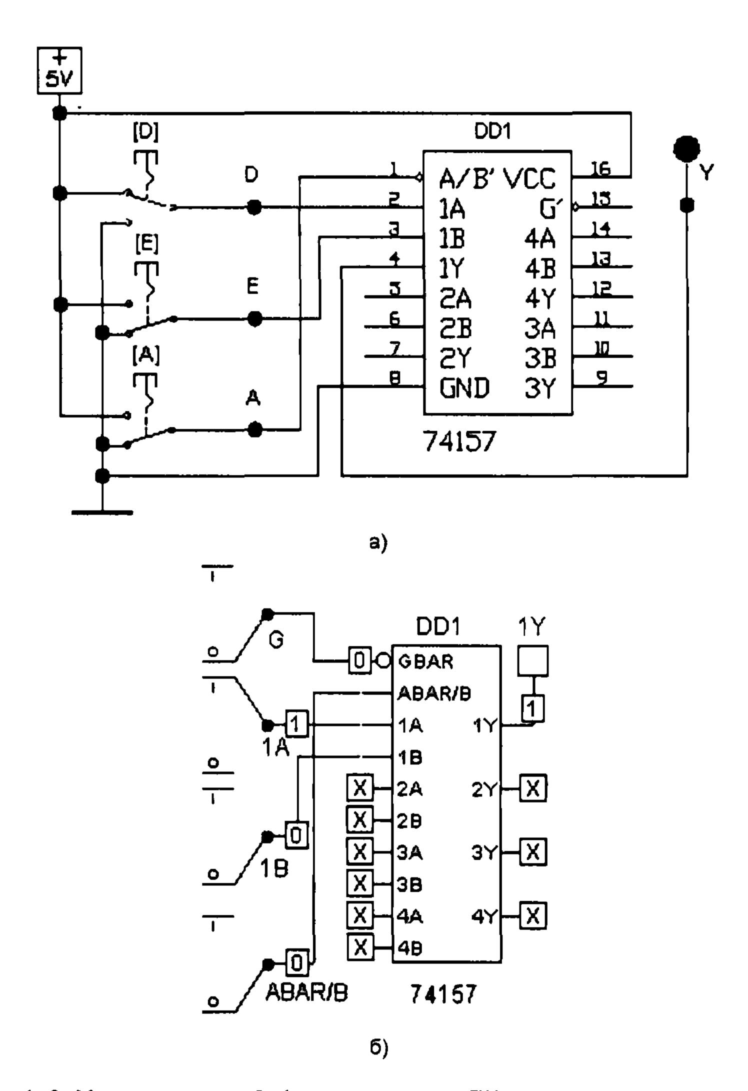


Рис. 103. Мультиплексор 2×1 на микросхеме ТТЛ 74157:
а – **EWB**, б – **MC**

Для преемственности с предыдущей схемой (см. рис. 102,в) на ключах сохранены прежние буквенные метки. На микросхеме DD1 корпусной вывод 1 является адресным входом, что обозначено на поле УГО микросхемы как A/B`. Дополнительный вход G`,

служит для подачи сигнала Strobe Стробом называют тактовый импульс от греч Strobes – вихрь, входящего в слово стробоскоп (греч Scopeo – смотрю) – прибор для прерывистого наблюдения в кратковременные моменты освещения (стробирования) объекта Вход строба является приоритетным управляющим входом разрешения если на нем установить 1, то на всех выходах будет 0 независимо от состояния других входов

В программе **MC** мультиплексор на микросхеме ТТЛ 74157 показан на рис. 103,б Здесь разрешающий вход обозначен через GBAR и на него надо подать низкий уровень напряжения Поскольку остальные выводы мы не задействовали, то на них появляется неопределенный сигнал, обозначенный как X (см рис 103, б) Для мультиплексора 4·1 состояние (в обозначениях микросхемы 74153) описывается таблицей:

B	A	Y
0	0	C_0
0	1	C_1
1	0	C_2
1	1	C_3

и, соответственно, уравнением:

$$Y= C_0A`B`+C_1AB`+C_2A`B+C_3AB.$$

Отсюда, по аналогии с предыдущим, нетрудно построить логическую структуру мультиплексора 4×1 (см. рис. 104,а)

Используя стандартную микросхему ТТЛ 74153, содержащую в корпусе два мультиплексора 4×1, также можно продемонстрировать его работу (см. рис. 104,б,в). Так, на рис 104,б на адресных входах набрано в двоичном коде число 3 и сигнал с входного канала 3 (равный 1) попадает на выход – выходной индикатор горит

Поскольку для создания мультиплексоров с большим числом входов требуются и БЛЭ *ИЛИ* с большим числом входов, то их создают каскадированием

Для создания мультиплексора 16×1 используем мультиплексоры 4×1, которые поместим в субблоки. Используя файл схемы на рис. 104,а, создадим субблок, который после графического редактирования показан в развернутом виде на рис. 105, а и в свернутом – на рис. 105,б.

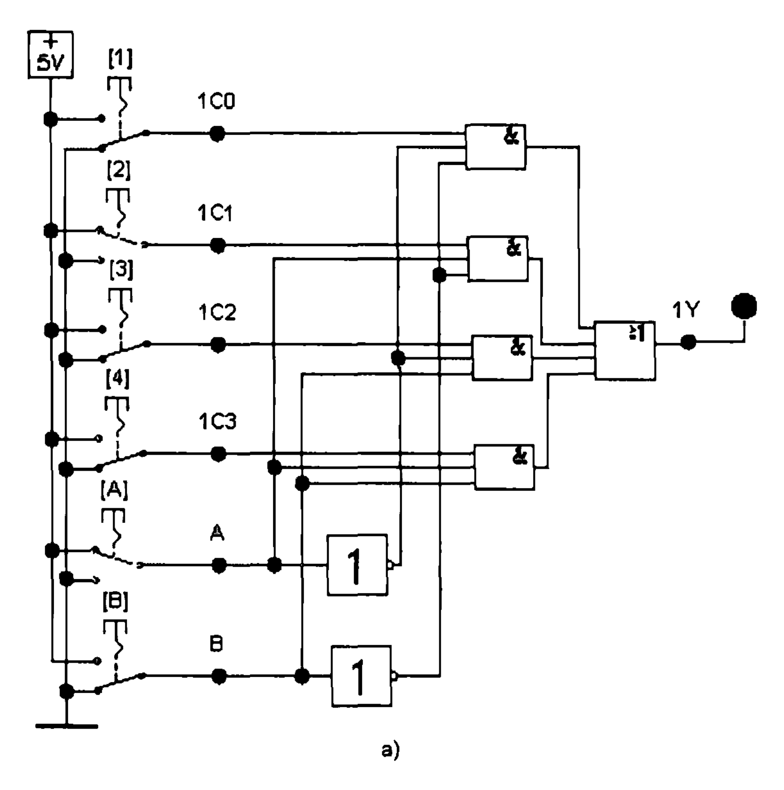


Рис 104 (начало)

Из пяти таких мультиплексоров, используя пирамидальное каскадирование, можно создать мультиплексор 16×1 (см рис. 106, а). Подобный мультиплексор имеется в виде микросхемы ТТЛ 74150 (см. рис. 106,б). Функционирование этого мультиплексора описывается ТИ, показанной на рис 106,в. Здесь при нормальной работе на вход разрешения надо подать низкий уровень (L – от англ. Low – низкий). При подаче на входы A .D четырехразрядного двоичного адреса на выходе появится сигнал с одного из входов E0–E15 в инверсной форме

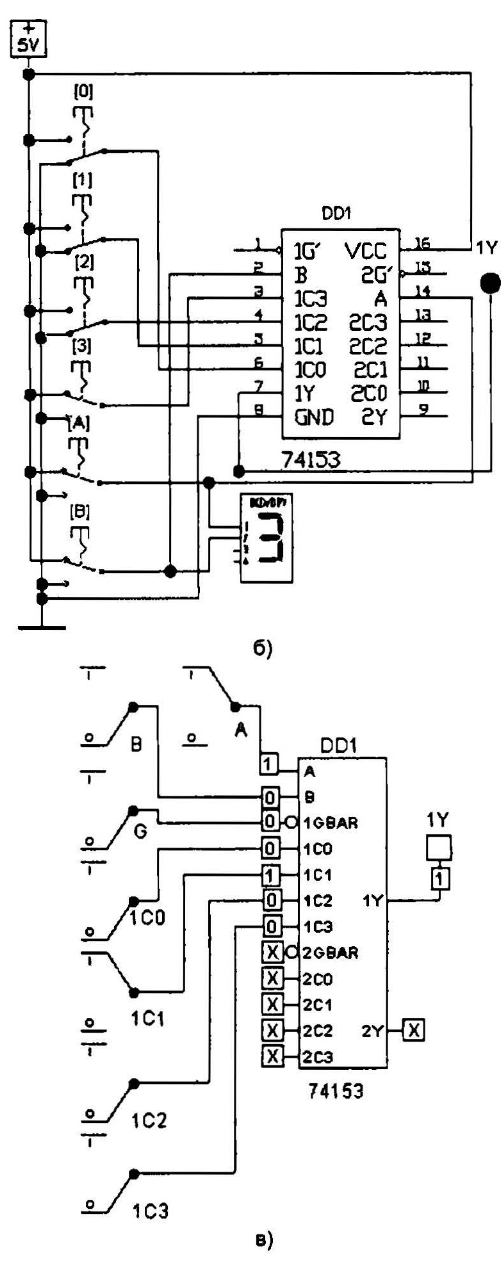
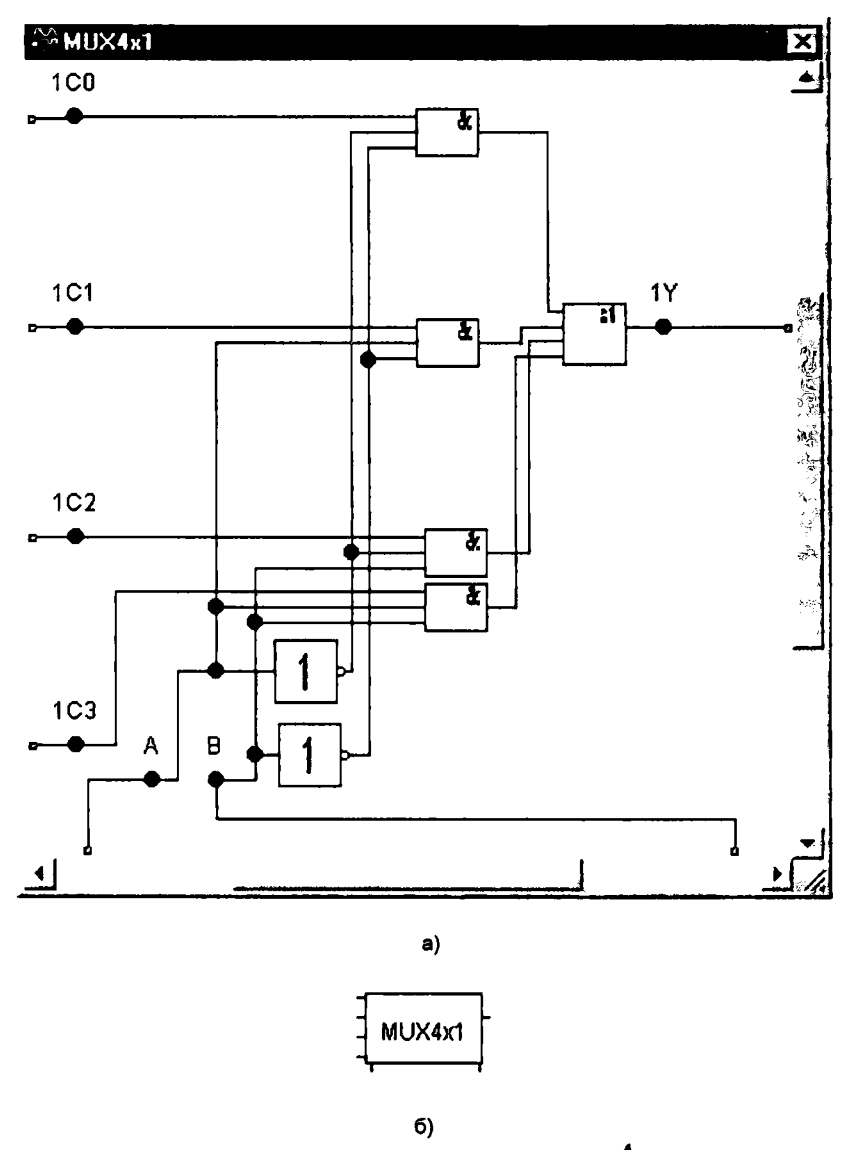


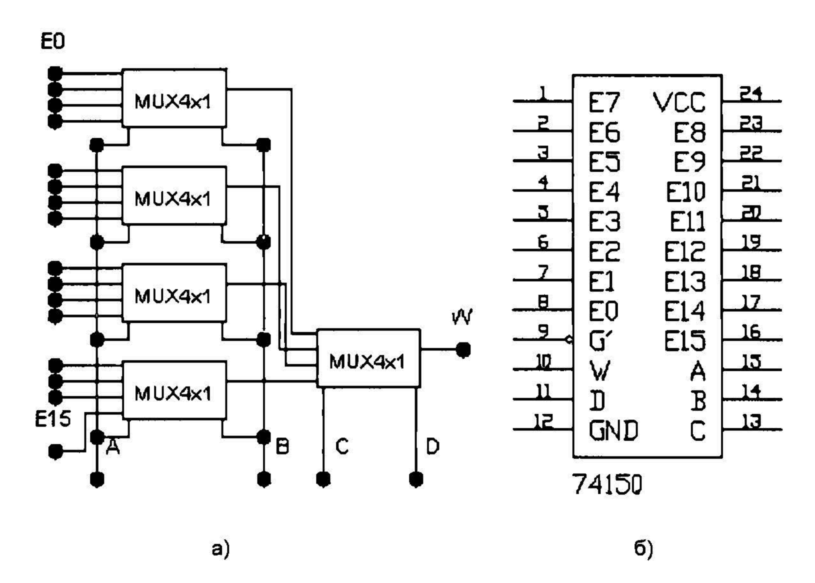
Рис 104 (окончание) Мультиплексор 4×1



а)

б)

Рис 105 Создание субблока мультиплексора 4×1 (EWB)



а) б)

74150 (1-of-16 Data Sel/MUX)					
Inputs					Output
D	B	C	A	G`	W
X	X	X	X	H	H
0	0	0	0	L	(EO)`
0	0	0	1	L	(E1)`
0	0	1	0	L	(E2)
0	0	1	1	L	(E3)`
0	1	0	0	L	(E4)`
0	1	0	1	L	(E5)
0	1	1	0	L	(E6)`
0	1	1	1	L	(E7)
1	0	0	0	L	(E8)`
1	0	0	1	L	(E9)`
1	0	1	0	L	(E1O)
1	0	1	1	L	(E11)
1	1	0	0	L	(E12)`
1	1	0	1	L	(E13)`
1	1	1	0	L	(E14)
1	1	1	1	L	(E15)`

в)

Рис 106 Мультиплексор 16×1 (**EWB**)

Демультиплексоры

Эти функциональные узлы выполняют обратную по отношению к мультиплексорам операцию над сигналами· цифровая информация, приходящая по одной линии, передается по выбору на одну из нескольких выходных линий. Тот же двухпозиционный ключ (Switch), используемый так, чтобы он имел один вход, который можно было бы соединять с одним из двух выходов является простейшим контактным демультиплексором. Выбор выходных линий в демультиплексорах осуществляется соответствующей адресацией в двоичном коде Работа демультиплексора, таким образом, тоже аналогична работе переключателя, только теперь он из одного входа коммутирует сигнал на любой из многих своих выходов В житейском смысле, мультиплексор делает выбор: «откуда взять», а демультиплексор – «куда отдать». На рабочем поле УГО микросхемы демультиплексора сокращенно пишут DEMUX (от англ. DEMULTIPLEXOR) или DMX.

Состояния простейшего демультиплексора 1×2, т е имеющего на один информационный вход D и два выхода Y_0 и Y_1 с адресацией по входу A, описываются следующей таблицей

A	Y_0	Y_1
0	D	0
1	0	D

Из данной таблицы:

Y_0=A'D и Y_1=AD.

Эти уравнения реализуются двумя элементами AND и одним инвертором (см рис 107)

Демультиплексор 1·4, имеющий на один информационный вход D четыре выхода Y_0 ...Y_3 с адресами A и B, описываются следующей таблицей:

B	A	Y_0	Y_1	Y_2	Y_3
0	0	D	0	0	0
0	1	0	D	0	0
1	0	0	0	D	0
1	1	0	0	0	D

По этой таблице составляем уравнения демультиплексора:

Y_0=A\`B\`D;
Y_1=A\`B D;
Y_2=A B\`D;
Y_3=A B D.

Отсюда видно, что для реализации демультиплексора 1×4 необходимо иметь четыре трехвходовых БЛЭ AND и два инвертора (см. рис. 108).

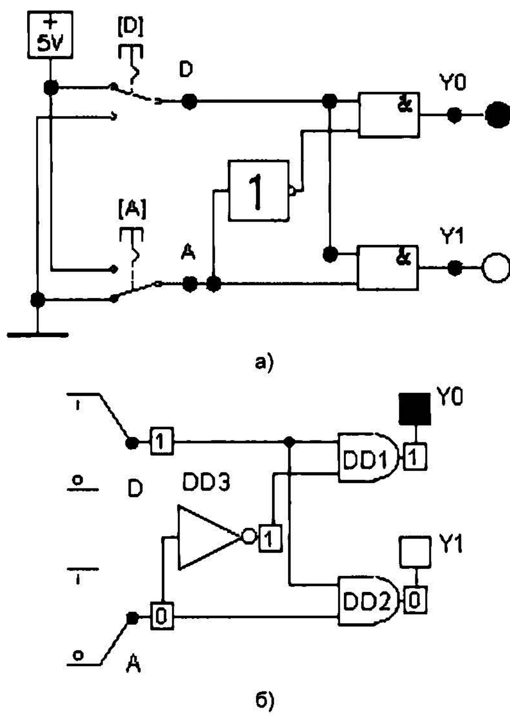


Рис. 107. Логическая структура демультиплексора 1×2: а – **EWB**; б – **MC**

Логические функции демультиплексора отличаются от логических функций аналогичного дешифратора только множителем D: при D=1, демультиплексор функционирует как дешифратор Поэтому вход разрешения микросхем дешифраторов может служить информационным входом демультиплексора Примером может служить микросхема ТТЛ 74139, имеющая в одном корпусе два дешифратора/демультиплексора. Таблица истинности этой

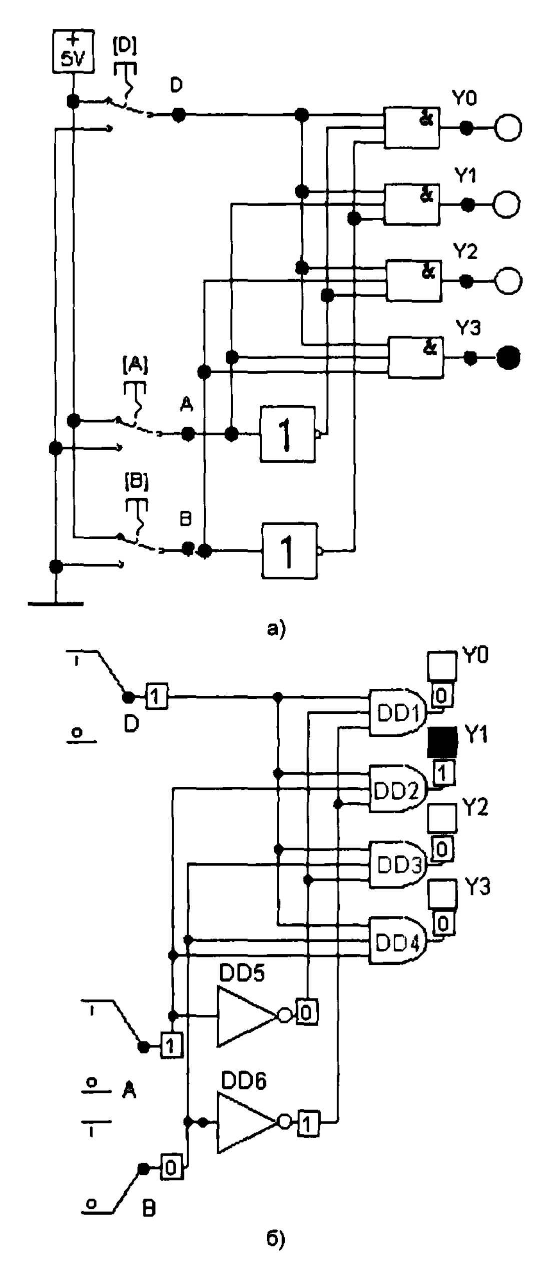


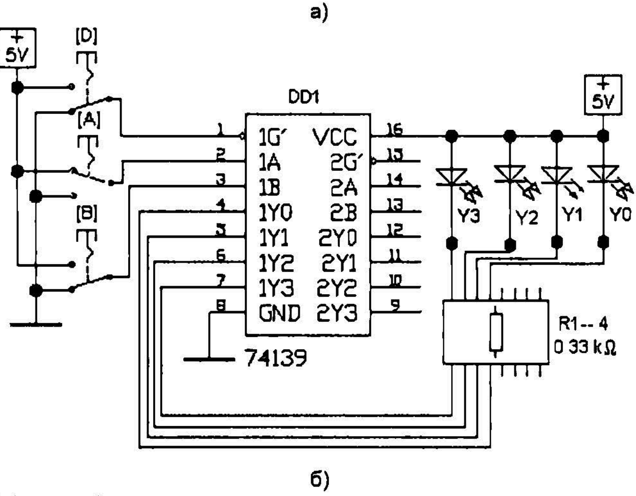
Рис. 108 Логическая структура демультиплексора 1×4: а – **EWB**; б – **MC**

74139 (Dual 2-to-4 Dec/DEMUX)

2-to-4 decoder/demultiplexer truth table

Enable $\bar{G}$	Select B	Select A	Outputs Y0	Outputs Y1	Outputs Y2	Outputs Y3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

а)

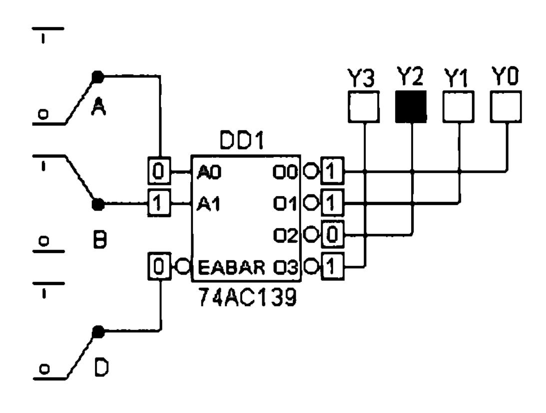


б)

Рис 109 (начало)

микросхемы показана на рис 109,а При подаче на разрешающий вход (Enable G`) лог. 1 выбранный выход также переводится в состояние 1, и наоборот

Сборка схем на микросхеме 74139 показана на рис 109, б, в В связи с тем, что выходы этой микросхемы инверсны, в программе **EWB** в качестве выходных индикаторов (как и ранее в подобных случаях) использованы светоизлучающие диоды, снабженные здесь еще токоограничивающими резисторами R1–R4 для корректности моделирования.



в)

Рис 109 (окончание) Демультиплексор 1×4 на микросхеме 74139
а – ТИ, б – **EWB**, в – **MC**

2.3. Компараторы

Вопрос мой прост и краток, –
Промолвил Носорог, –
Что лучше – со'рок пя'ток
Или пя'ток соро'к? –
Увы, никто на это
Ответа
Дать не мог!
А.А. Милн.
Винни-Пух и все-все-все

При обработке информации в цифровой форме часто встречаются операции сравнения чисел по различным признакам: по модулю; сравнение с учетом знаков чисел, сравнение порядков и мантисс чисел, представленных в форме с плавающей запятой. Эти операции используются в счетчиках, дешифраторах и сумматорах, в блоках микропрограммного управления, стабилизаторах, в кэш-памяти и т.п.

Цифровые компараторы вырабатывают сигналы отношений типа «меньше», «меньше или равно», «равно», «больше или равно», «больше» между двумя числами (кодами).

Название «компаратор» происходит от латинского слова comparator, означающего «сравнивающий». Изначально компараторы как физические приборы использовались для сравнения штриховых и концевых мер длины с эталоном (например, знаме-

нитым парижским эталоном метра) Впоследствии термин «компаратор» закрепился и за электронными приборами, осуществляющими функцию сравнения различных величин.

Рассмотрим простейший компаратор, с помощью которого можно сравнить два числа A и B одинаковой разрядности, заданных в двоичном или двоично-десятичном коде. В зависимости от поставленной задачи компараторы могут определять три различных функции: $F_=$, если A=B, $F_<$, если A<B и $F_>$, если A>B В соответствии с этим таблица истинности компаратора одноразрядных кодов имеет вид.

A	B	$F_=$	$F_<$	$F_>$
0	0	1	0	0
0	1	0	0	1
1	0	0	1	0
1	1	1	0	0

Анализ этой таблицы показывает, что для определения равенства двух величин ($F_=$) надо использовать рассмотренную ранее ЛФ типа *Исключающее ИЛИ-НЕ* (XNOR) Два других случая соответствуют известным ЛФ типа «А, но не В» (антисовпадение) и «не А, но В» (обратное антисовпадение) Для получения логической структуры компаратора воспользуемся логическим конвертером. Вводя в ЛК три раза ТИ (отличающиеся выходным столбцом), получим соответствующие формулы и схемы На рис 110,а показана раскрытая лицевая панель логического конвертера с установками для первого случая Далее на рис. 110,б,в,г, показаны соответственно схемы для всех трех случаев. Заменив схему на рис. 110,б на БЛЭ XNOR и дополнив схемами, показанными на рис. 110,в,г, получим логическую структуру компаратора в программе **EWB** (см. рис. 111,а) и в программе **МС** (см. рис. 111,б). Задавая в этих схемах различные сигналы и включая моделирование, можно убедиться, что они работают в соответствии с ТИ компаратора, приведенной выше

В программе **МС** проведем испытание микросхемы 74520, которая является восьмиразрядным компаратором с инверсным выходом (рис. 112). Если на вход разрешения GBAR подать низкий уровень (EN=0), то при равенстве слов, подаваемых на входы P0–P7 и Q0...Q7, на выходе установится низкий уровень $F_= =0$. Так на рис. 112 показано, что при единице на всех информационных входах, кроме P4=Q4=0, выход равен 0 (слова равны). Доста-

точно поменять один бит в сигналах, чтобы они различались, и перезапустить моделирование, как на выходе появится 1, что при инверсном выходе означает неравенство входных слов Характер этого неравенства данной микросхемой не определяется

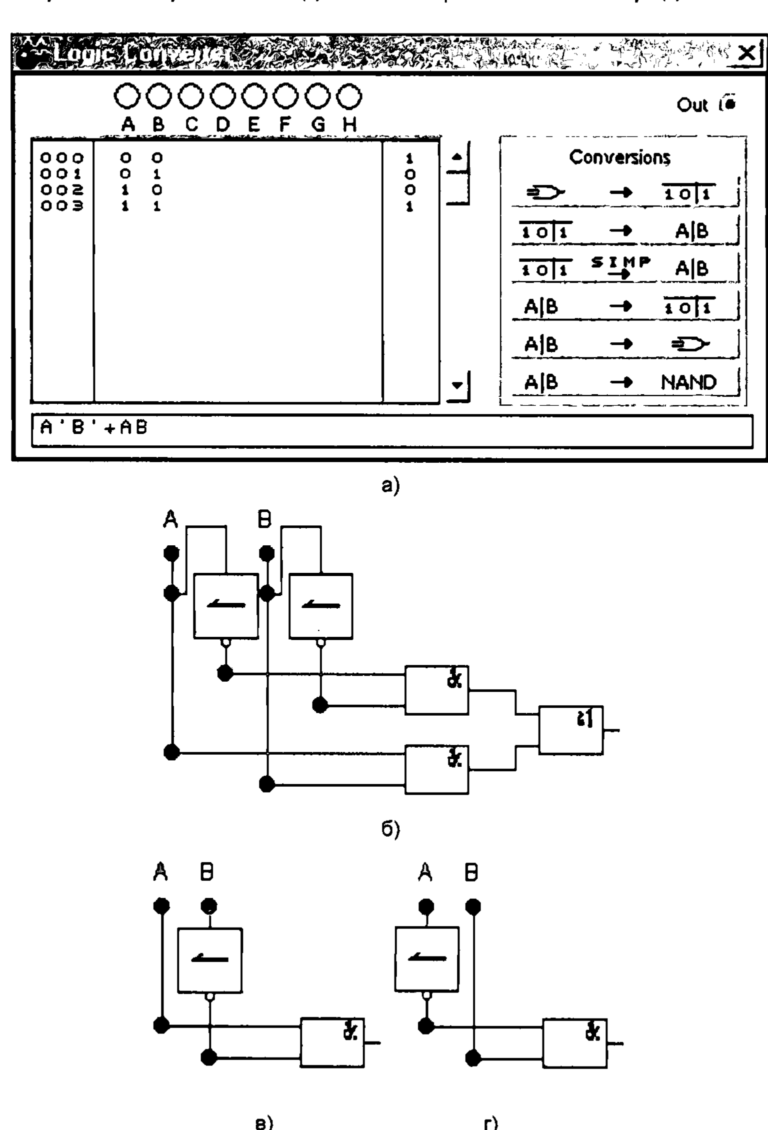


Рис. 110. Разработка структуры компаратора (**EWB**)

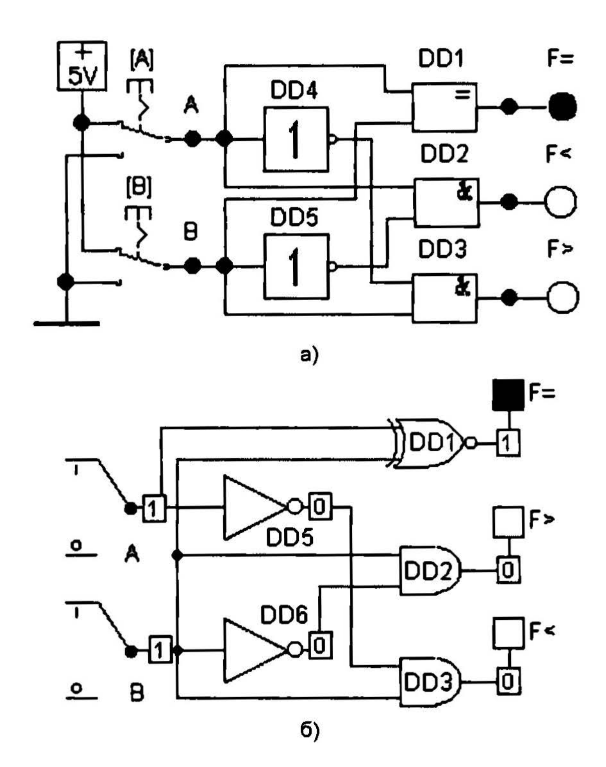


Рис 111. Логическая структура компаратора: а – **EWB**; б – **MC**

2.4. Сумматоры

– Всего, – возглесил Остап, – четыреста восемьдесят восемь рублей Эх! Двенадцати рублей не хватает для ровного счета
И. Ильф, Е. Петров. Двенадцать стульев

Полный сумматор

Рассмотренный выше полусумматор применим лишь для сложения одноразрядных двоичных кодов или самых младших значащих разрядов многоразрядных слов. При сложении старших разрядов многоразрядных двоичных слов необходимо в схеме учесть перенос из предыдущего (младшего) разряда. Такое уст-

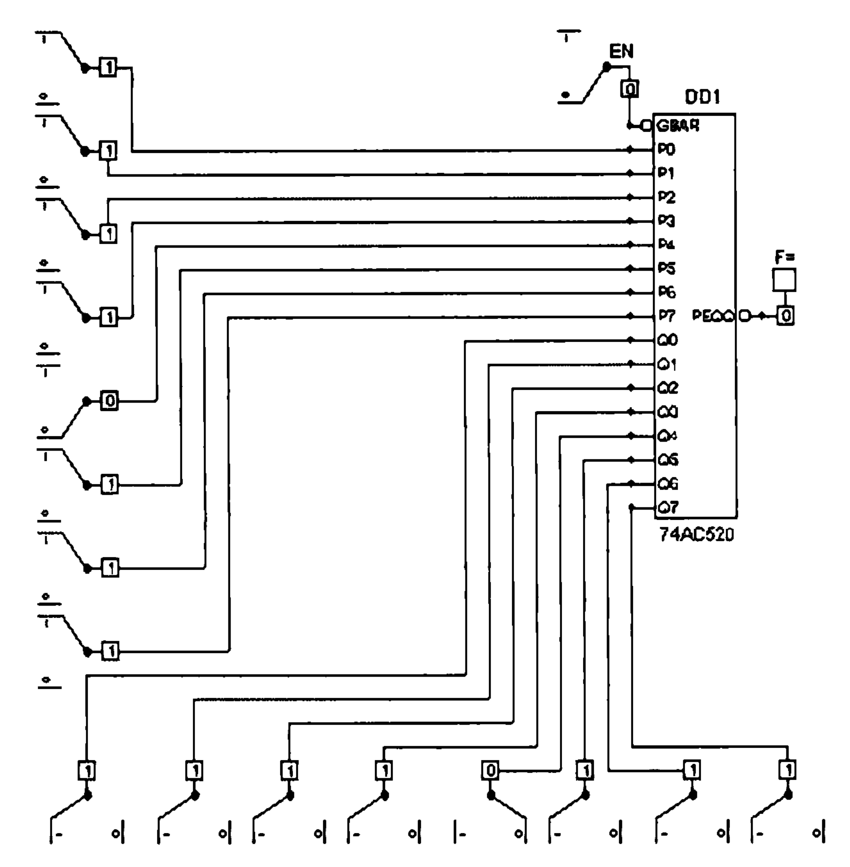


Рис. 112. Компаратор на микросхеме 74520 (**МС**)

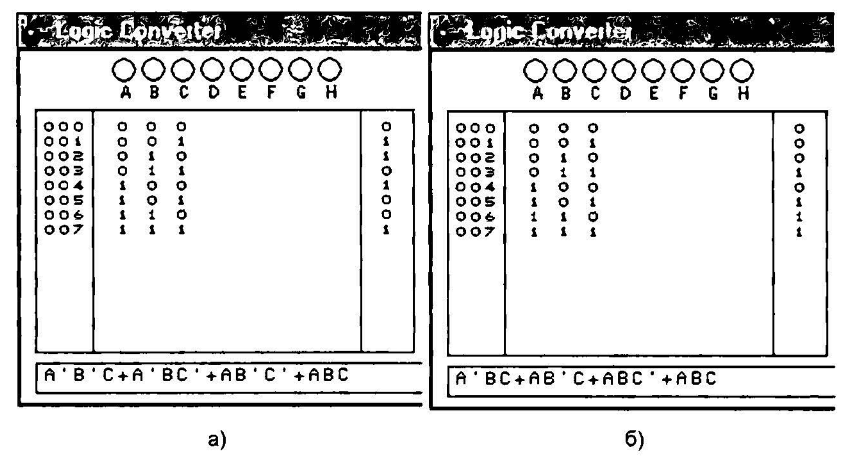
ройство является полным сумматором: оно должно иметь три входа (A, B и C_{in} – вход переноса) и два выхода: S – суммы и C_{out} – переноса

Составим ТИ для S и C_{out} полного сумматора согласно логике его работы и правилам сложения двоичных чисел. Эти ТИ занесем на панель ЛК и там же найдем их ЛФ (рис. 113,а,б).

Перепишем ЛФ с экранной строки, введя принятые выше обозначения:

$S=A'B'C_{in}+A'BC_{in}'+AB'C_{in}+' \quad _{in};$

$C_{out}=A'BC_{in}+AB'\,C_{in}+ABC_{in}'+ABC_{in}.$



а) б)

Рис 113. ТИ и ЛФ полного сумматора: а – S, б – Cout (**EWB**)

После тождественных преобразований эти ЛФ принимают вид:

$S=(A \oplus B) \oplus C_{in}$;

$C_{out}=AB+C_{in}(A \oplus B)$.

Анализируя последние выражения, видим, что над А и В надо произвести операцию XOR и полученный результат использовать два раза: еще раз XOR с C_{in} – это даст S, потом операция AND с C_{in}, а с ее результатом операцию OR с AB (которое получается как перенос из первого полусумматора), и это даст перенос C_{out}. Таким образом, для построения полного сумматора требуется 5 ЛЭ 2 типа XOR, 2 типа AND и 1 типа OR. Выведем их на рабочее поле и соединим в соответствии с приведенными ЛФ (рис. 114,а,б).

Задавая различные значения входным сигналам, можно наблюдать за работой сумматора. В программе **MC** при моделировании показываются логические состояния всех цифровых узлов на схеме, что позволяет глубже познакомиться с работой устройства. ЛЭ DD1,DD2 и DD3,DD4 на этих схемах представляют два полусумматора (HA1 и HA2), а через ЛЭ DD5 осуществляется вывод итогового переноса. Таким образом, используя два полусумматора и ЛЭ OR, можно построить полный сумматор. Это продемонстрированно в программе **EWB** на рис. 115,а. В этой же программе имеется и готовая библиотечная модель такого типа Full

Adder (FA – полный сумматор), и ее можно выбрать по пиктограмме [Σ CI CO] в панели Digital. УГО этого элемента (в отличие от пиктограммы) имеет, как и положено, три входа и два выхода Заменив полусумматоры HA1, HA2 и ЛЭ OR на полный сумматор FA, получаем соответствующую схему, показанную на рис. 115,б.

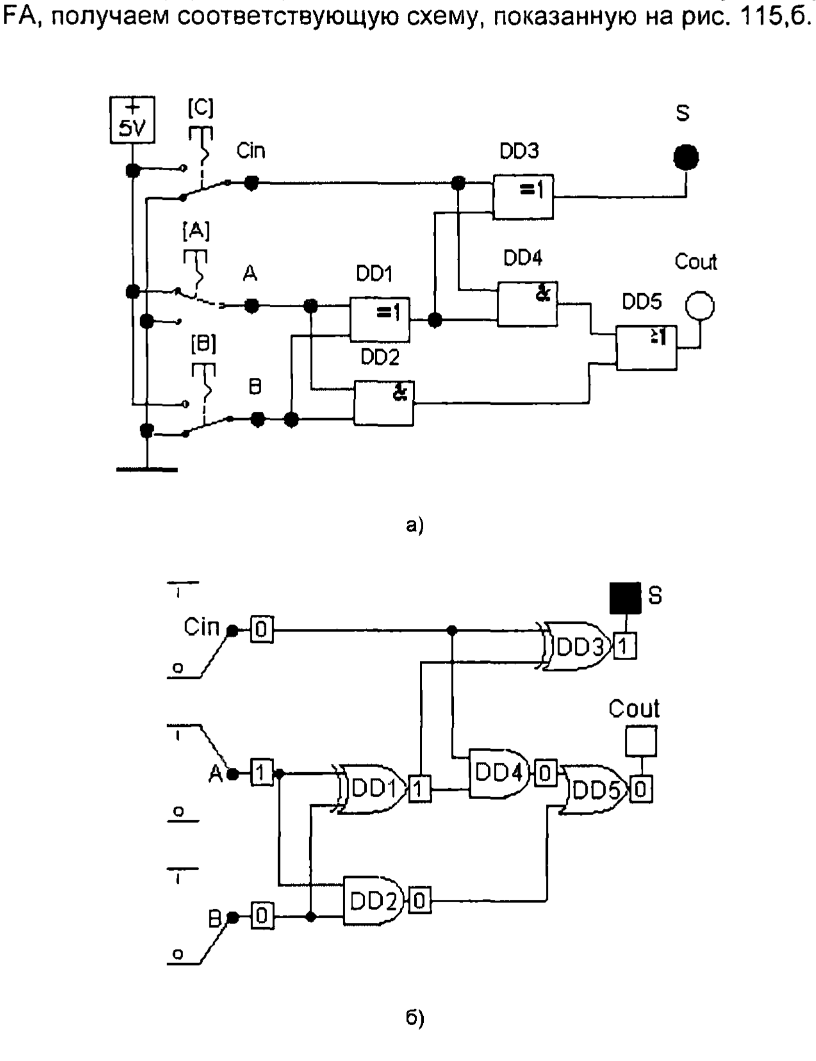


Рис 114 Логическая структура полного сумматора: а – **EWB**, б – **MC**

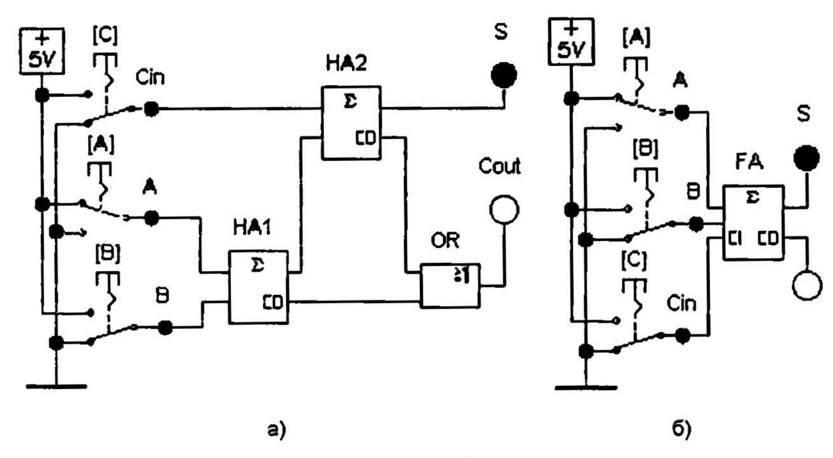


Рис 115 Модель полного сумматора (**EWB**)

Сумматоры на интегральных микросхемах

Некоторые типы сумматоров выпускаются в интегральном исполнении и являются однокристальными устройствами В качестве примера рассмотрим работу КМОП микросхемы 4008, являющейся четырехразрядным полным сумматором Она состоит из четырех одноразрядных сумматоров и схемы ускоренного параллельного переноса.

В программе **EWB** соберем на основе этой микросхемы сумматор (см. рис. 116). Занесение в данный сумматор двух четырехразрядных слагаемых $A_3A_2A_1A_0$ и $B_3B_2B_1B_0$ производится ключами [1]...[8]. Для удобства отсчета вводимых и выводимых двоичных чисел использованы семисегментные индикаторы с декодерами (A, B и SUM).

На входе переноса из предыдущего разряда CIN добавлен ключ [I]. Этот вход и выход переноса COUT контролируются логическими пробниками. На рис. 116 зафиксировано сложение:

A=3, B=6, CIN=1.

На выходе сумматора SUM=A, т е. число 10 при COUT=0.

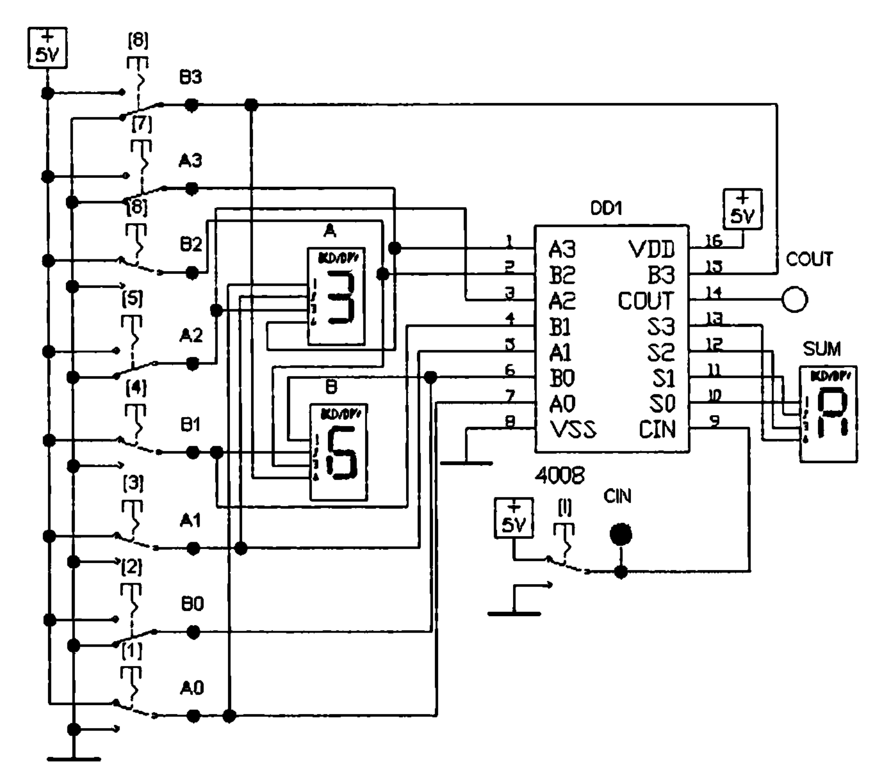


Рис 116. Полный сумматор на микросхеме 4008 (**EWB**)

Совет Прежде чем проводить суммирование чисел, проверьте работу схемы, подавая вначале только на входы $A_3A_2A_1A_0$ последовательно все двоичные числа от 0 до 15, затем, аналогично, на входы $B_3B_2B_1B_0$ При наблюдении чисел на семисегментном индикаторе сверяйтесь с рис. 101,в и обозначениями чисел в разных кодах

Сумматоры на микросхемах можно соединять друг с другом, подключая выход COUT предыдущей микросхемы к входу CIN – последующей Так, из двух микросхем 4008 можно составить восьми разрядный двоичный сумматор

В программе **МС** также рассмотрим четырехразрядный полный двоичный сумматор, но на основе микросхем ТТЛ 74283 Воспользуемся для этого библиотечным файлом 283 cir, заменив

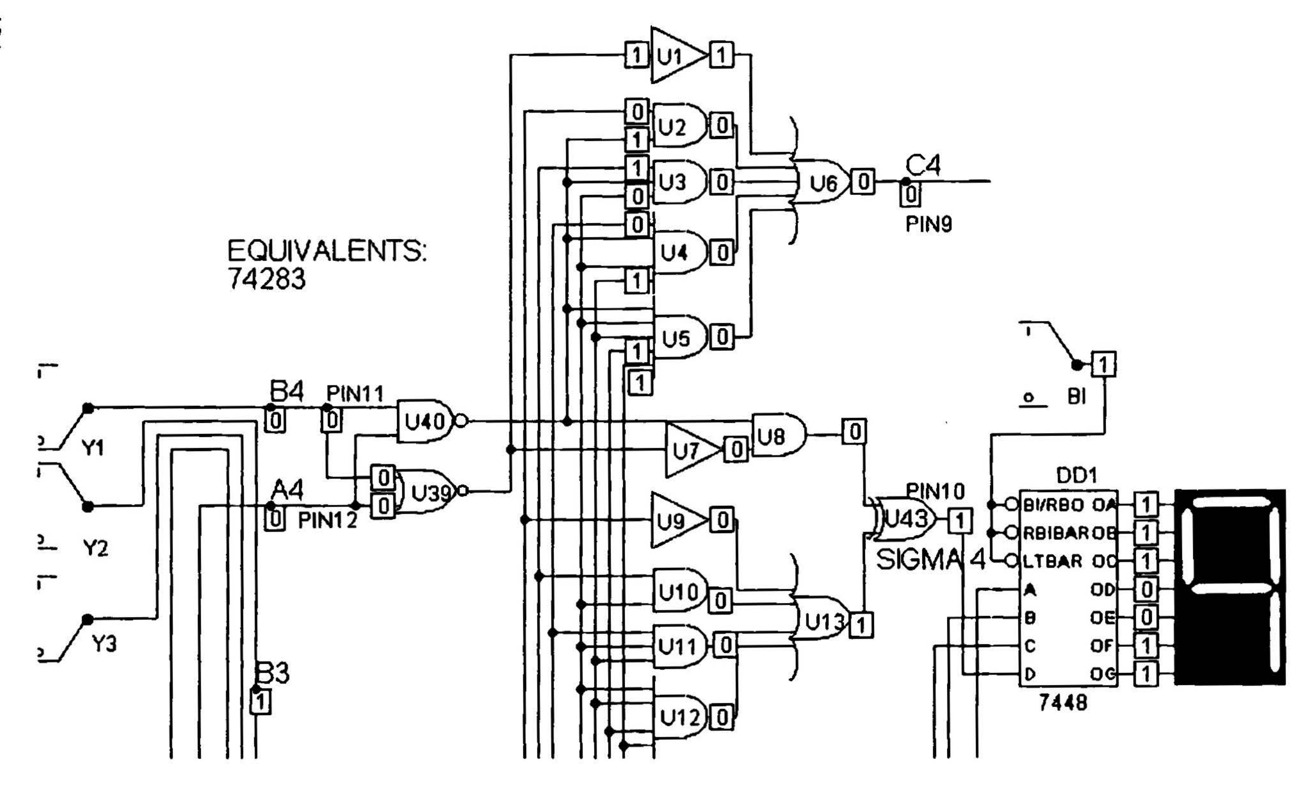


Рис 117 (начало)

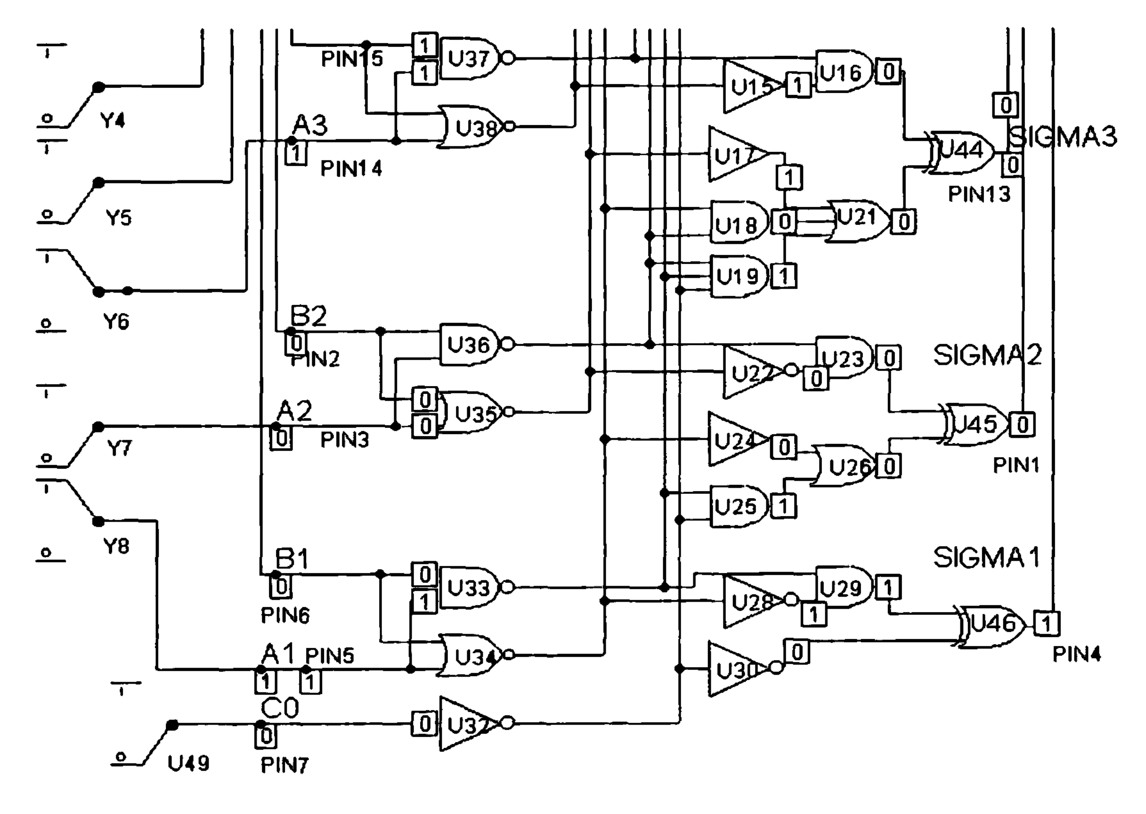


Рис. 117 (окончание). Логическая структура полного сумматора микросхемы 74283 (**МС**)

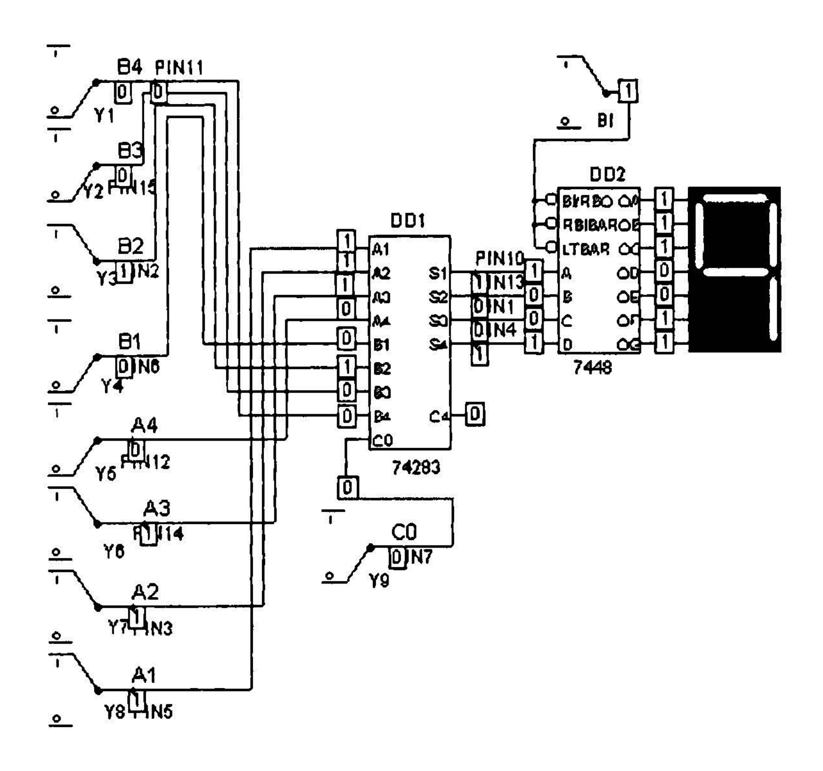


Рис 118. Полный сумматор на микросхеме 74283 (**МС**)

источники входных сигналов на цифровые ключи и дополнив схему семисегментным индикатором с ранее рассмотренным драйвером на микросхеме 7448 (см. рис. 117)

Здесь представлена развернутая логическая структура интегральной микросхемы сумматора В этой структуре видны (как на этажерке) четыре одноразрядных полных двоичных сумматора и блок ускоренного параллельного переноса (в самой верхней части схемы) При работе сумматора первый операнд (компонент проводимой операции в данном случае слагаемое) подается на входы А1 – А4 младший разряд на А1, старший – на А4. Второй операнд подается на входы В1 – В4 младший разряд на В1, старший – на В4 Сумма обоих чисел формируется на выходах SIGMA1 – SIGMA4, младший разряд на SIGMA1, старший – на SIGMA4 Когда результат суммирования превысит в десятичной системе число 15 (в двоичной системе соответственно – 1111), на

выходе С4 появится напряжение высокого уровня. Вход сигнала переноса С0 при использовании только четырехразрядных чисел соединяется с общим проводом. Если же используется восьмиразрядное число (четыре старших разряда), то вход С0 соединяется с выходом С4 предшествующей ступени (младшие разряды) Метками Pin1... Pin13 обозначены выводы на корпусе реальной микросхемы. На показанном рисунке введены двоичные числа 0101 и 0100 (5 и 4 в десятичной системе) и получен результат 9

Этот же сумматор, но с микросхемой в свернутом виде, показан на рис 118

3. УЗЛЫ ПОСЛЕДОВАТЕЛЬНОСТНОГО ТИПА

3.1. Триггеры

Английское слово «триггер» означает спуск затвора оружия с боевого взвода Спуск вызывает выстрел, и в этом значении (как «спускатель») понятие триггера перешло в электронику Триггер всегда чем-то управляет на старте, что-то запускает

Й.Янсен. Курс цифровой электроники

Асинхронный RS-триггер

Если внимательнее присмотреться к схемам (рис. 80,а и 81,а), то нетрудно увидеть в них то, что по отношению ко второму выходу (Out2) также действует обратная связь Перечертим схему на рис 81,а с учетом этого обстоятельства (см рис 119,а). Кроме того, в схеме можно увидеть и симметрию, если ее изобразить так, как показано на следующем рисунке (см рис. 119,б)

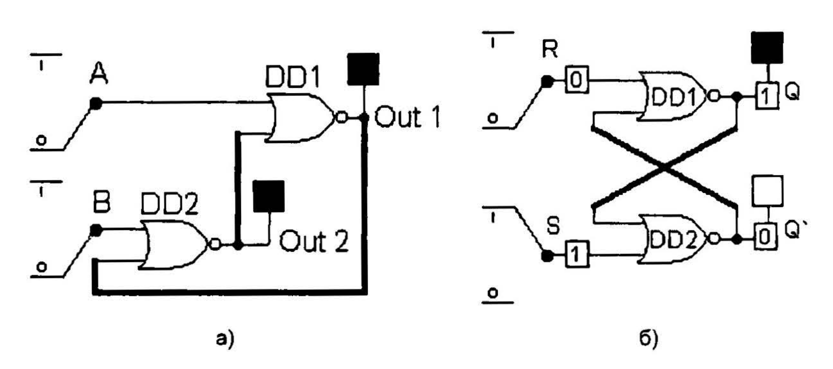


Рис. 119 Схема RS-триггера (**МС**)

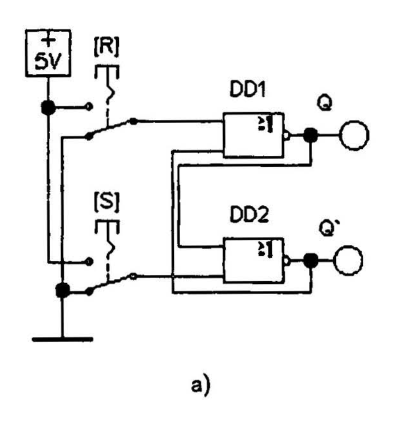
Здесь отчетливо видны перекрестные положительные обратные связи, а выходы и входы получили свое законное общепринятое обозначение R – Reset (сброс в 0), S – Set (установка в 1), Q – Quit (выход) и Q` – инверсный выход Поскольку схема симметрична, то можно поменять местами обозначения входов, но тогда одновременно надо поменять местами и обозначения выходов Единого расположения информационных выводов и их буквенных меток (R или S, сверху или снизу), а также последовательности букв в названии (RS или SR) не установлено Описанное цифровое устройство относится к одному из основных цифровых элементов ЭВМ – к триггерам

Как уже было указано, слово «триггер», по-английски trigger, означает спусковое устройство, курок, собачка, защелка В англоязычной технической литературе этот термин обычно используют в значении глагола «запускать», а собственно устройство в зависимости от его вида называют по-иному. Например, Flip – Flop (щелчок – хлопок), и иногда именно этот термин в англоязычной литературе вообще служит синонимом слова «триггер»

История триггеров началась с 1918 г. В то время они выполнялись на электронных лампах. М.А Бонч-Бруевич описал схему «катодного реле» с разрывными характеристиками. Практическая схема с двумя устойчивыми состояниями была опубликована английским физиком Уильямом Генри Икклзом (W H. Eccles) и неким Ф.В. Джорданом (F W Jordan). Последнюю схему называют триггером Икклза – Джордана.

Триггер, или бистабильный мультивибратор, – одноразрядный элемент памяти, предназначенный для хранения логической переменной Триггер относится к устройствам последовательностного типа, в котором выходной сигнал зависит от сигналов на входе как в текущий, так и в предыдущий момент времени

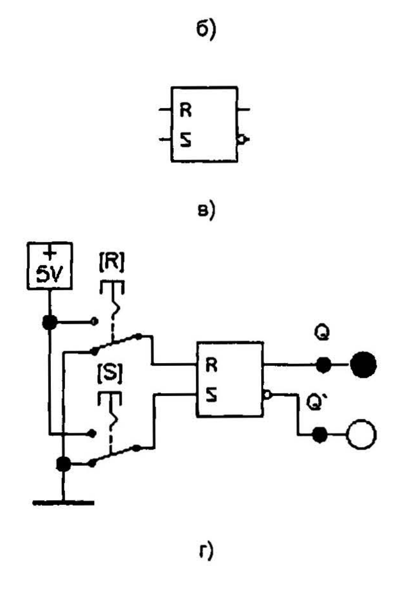
Рассмотрим RS-триггер в программе **EWB** (рис 120,а) Состояние триггера обычно отождествляют с сигналом на прямом выходе при Q=1 – единичное состояние (Q`=0) и при Q=0 – нулевое состояние (Q`=1) Задавая входам различные значения, проследим за состояниями триггера, учитывая также состояния в предыдущие моменты времени (для этого в программе **EWB** не надо выключать моделирование). Начнем с R=0, S=1 – Q=1, Q`=0, это установка единичного состояния, затем перейдем к R=0, S=0 – Q=1, Q`=0, это режим хранения предыдущего состояния, далее возьмем R=1, S=0 – Q=0, Q`=1, это сброс в нулевое состояние, при R=0, S=0 – Q=0, Q`=1, это опять режим хранения предыдущего состояния. На этом исчерпываются возможные рабочие со-



а)

R	S	Q	Q`
0	0	Qхран	Q`хран
0	1	1	0
1	0	0	1
1	1	...	...

б)



в)

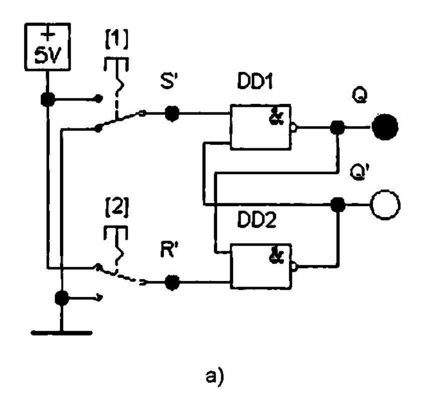
г)

Рис. 120. RS-триггер (**EWB**)

стояния, так как комбинация входных сигналов R=1, S=1 приводит к неопределенности и является для RS-триггера запрещенной Полученные результаты занесем в таблицу состояний (переходов), аналогичную таблице истинности для комбинационных схем (см рис 120,б) Переброс триггера из единичного состояния в нулевое, и наоборот, напоминает поведение детских качелей из доски с опорой посередине то один, то другой ребенок занимает высокий уровень. Качели, остановившиеся в положении «один из детей сверху, а другой – снизу» соответствуют режиму хранения.

На принципиальных схемах, а в нашем случае и на виртуальных схемных моделях, RS-триггер обозначают прямоугольником с соответствующей разметкой выводов В программе **EWB**, нажав на пиктограмму Digital, можно выбрать триггер (см рис 120,в), который в данной программе назван RS Flip-Flop В УГО рассматриваемого RS-триггера (см. рис 120,в) инверсный выход, как и положено, отмечен кружком С данной моделью можно провести все описанные выше эксперименты, собрав схему в соответствии с рис. 120,г.

Рассмотренный выше RS-триггер переключался сигналами лог 1 и поэтому относится к устройствам с прямым управлением или активным высоким уровнем. Триггер на двух элементах *И-НЕ* имеет инверсные входы R`, S` (см рис 121,а) и его переключения осуществляются при подаче на входы лог. 0 (см рис. 121,б) В этом случае активным является низкий уровень, а управление является инверсным Логическая структура R`S`-триггера в программе **MC** показана на рис 122



а)

Рис 121 (начало)

R	S	Q	Q`
0	0	...	..
0	1	0	1
1	0	1	0
1	1	Qхран	Q`хран

б)

Рис. 121 (окончание) R`S`-триггер (**EWB**)

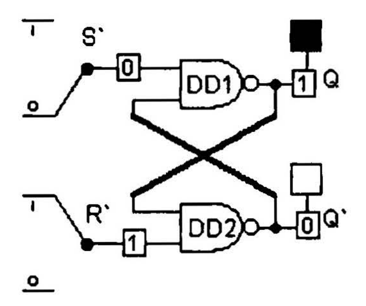
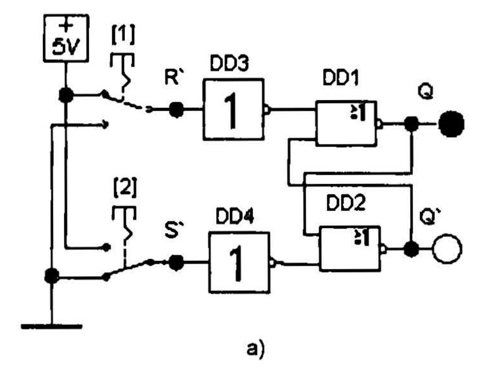


Рис. 122 R`S`-триггер (**MC**)



а)

Рис 123 (начало)

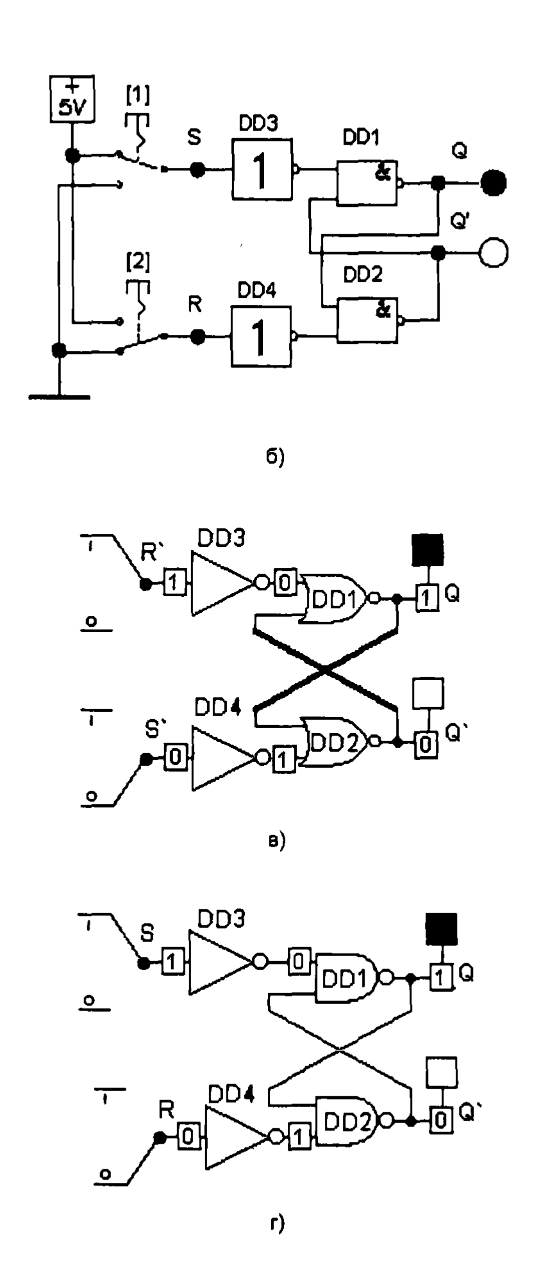


Рис 123 (окончание) Логические структуры RS и R`S`-триггероа с инверторами

Добавление инверторов во входные цепи рассмотренных триггеров меняет характер управления на противоположный (см рис 123,а-г) Поскольку ЛЭ *И-НЕ* (NAND) и *ИЛИ-НЕ* (NOR) легко превращаются в инверторы, то соответствующие триггеры можно собрать на одной элементной базе, например, на рис 124,а,б показаны логические структуры RS-триггеров на ЛЭ NAND

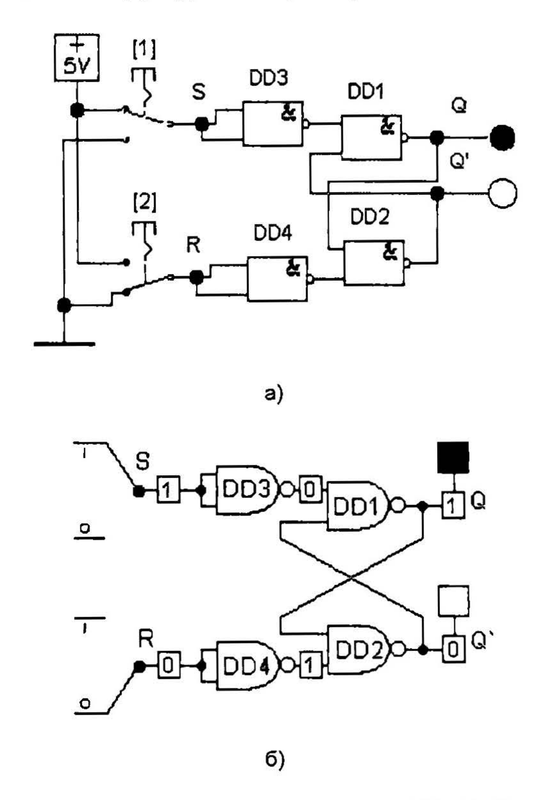


Рис 124 Логические структуры RS-триггеров на ЛЭ NAND

Рассмотренные триггеры относятся к асинхронному типу, поскольку моменты переключения никак не связаны с тактирующими сигналами управления В большинстве серий микросхем подобные триггеры как самостоятельные изделия отсутствуют, так как они легко собираются из БЛЭ.

Синхронный RS-триггер

Отличительной особенностью последовательностных логических устройств является возможность синхронизации их переходов с тактирующими импульсами. В отсутствие этих импульсов информационные входы как бы отключены Различают синхронные триггеры с управлением уровнем и с динамическим управлением

Синхронный RS-триггер с управлением уровнем можно получить из предыдущих схем слегка видоизменив их входную логику и дополнив ее синхронизирующим входом, обычно обозначаемым буквой С (от Clock Pulse – тактовый импульс).

На рис 125,а,б показаны схемы синхронных триггеров, полученные соответственно из схем рис 124,а,б В схеме на рис 125,в использована библиотечная модель RS-триггера (элемент DD3) Приведенные схемы типичны с точки зрения их состава для последовательностных устройств: в них присутствуют элементы комбинационной логики и ячейки памяти. Для того чтобы более подробно исследовать работу синхронного триггера, заменим в схеме на рис 124,б ключи на источники сигналов Stim1 (см рис 124,г), задав их в следующем формате.

```
define IN1 +0NS 0 +250NS 1 +650NS 0
define IN2 +0NS 0 +100NS 1 +200NS 0 +300NS 1 +400NS 0
+500NS 1 +600NS 0
+700NS 1 +800NS 0 +900NS 1 +1000NS 0 +1100NS 1 +1200NS 0
+1300NS 1 +1400NS 0
define IN3 +0NS 0 +850NS 1
```

Здесь IN1 соответствует входу S, IN2 соответствует входу С и IN3 соответствует входу R Результат анализа показан на графиках рис 125,г Видно, что результат соответствует таблице, показанной на рис 120,б с той особенностью, что переходы триггера происходят только при С=1.

Добавление в основную триггерную ячейку DD1–DD2 дополнительных асинхронных входов S` и R`, обладающих приоритетом, позволяет производить предварительную установку состояния триггера в 1 или сбрасывать его в 0 (см рис. 126,а,б)

Последние схемы легко выполняются из предыдущих заменой двухвходовых ЛЭ *И-НЕ* на 3-входовые (3-Input NAND Gate или 3- NAND) и добавлением установочных ключей S` и R`, как это показано на рис 126,а,б

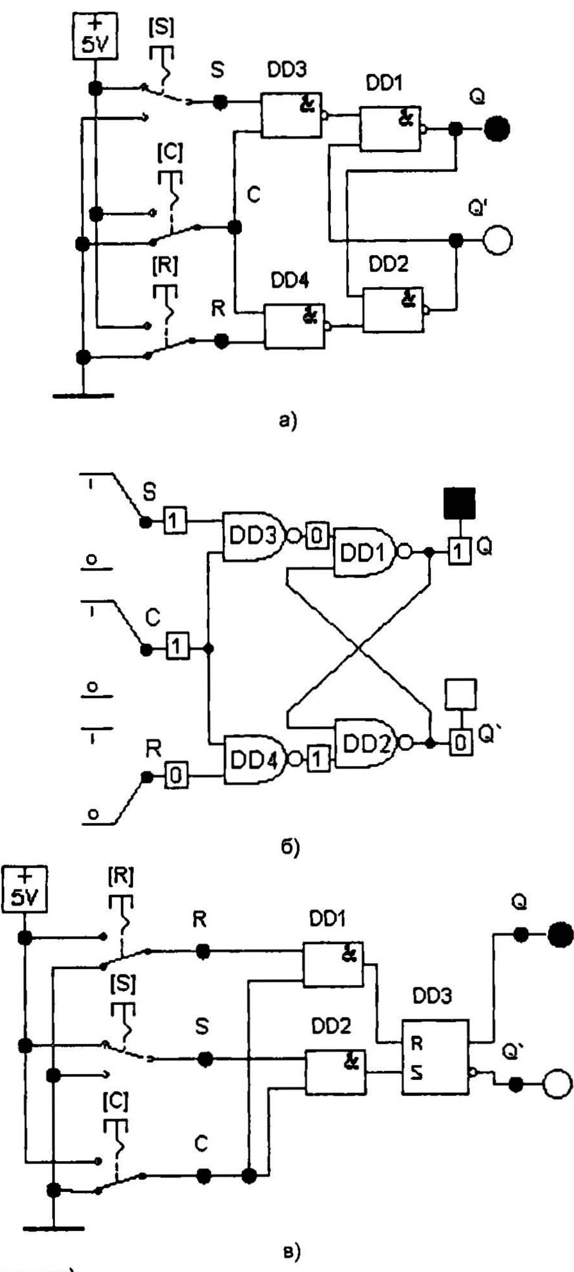


Рис 125 (начало)

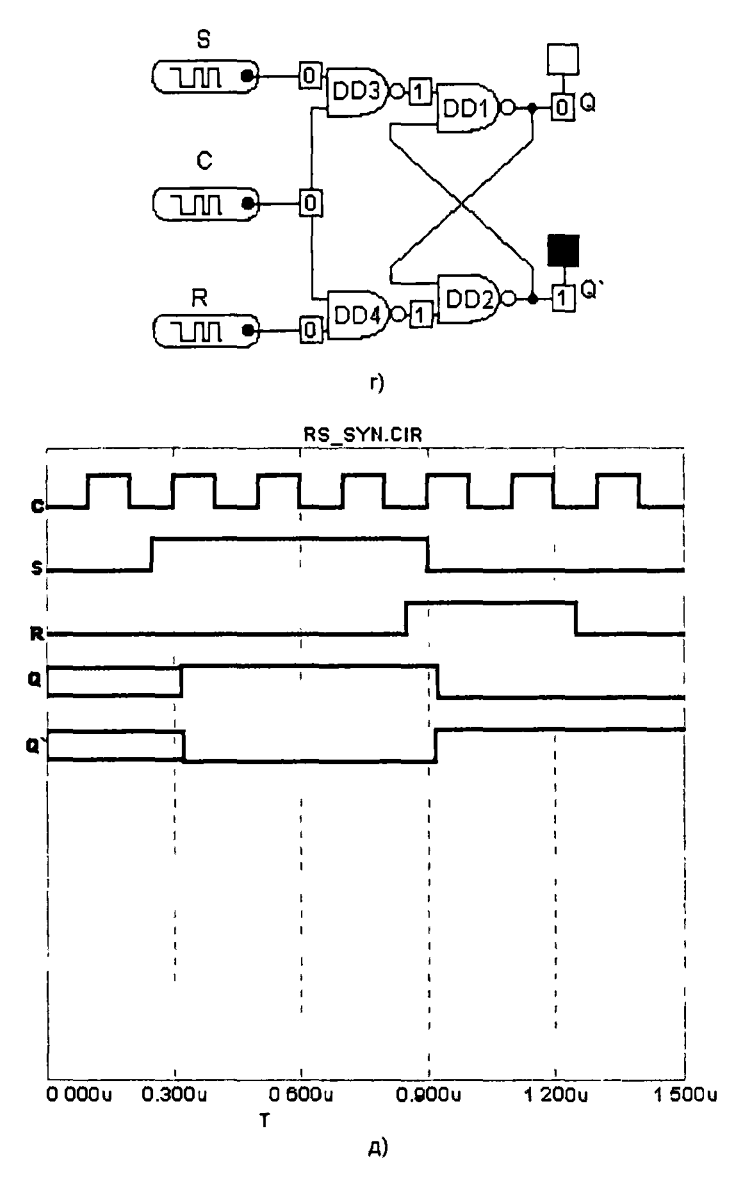


Рис 125 (окончание). Синхронный RS-триггер с управлением уровнем

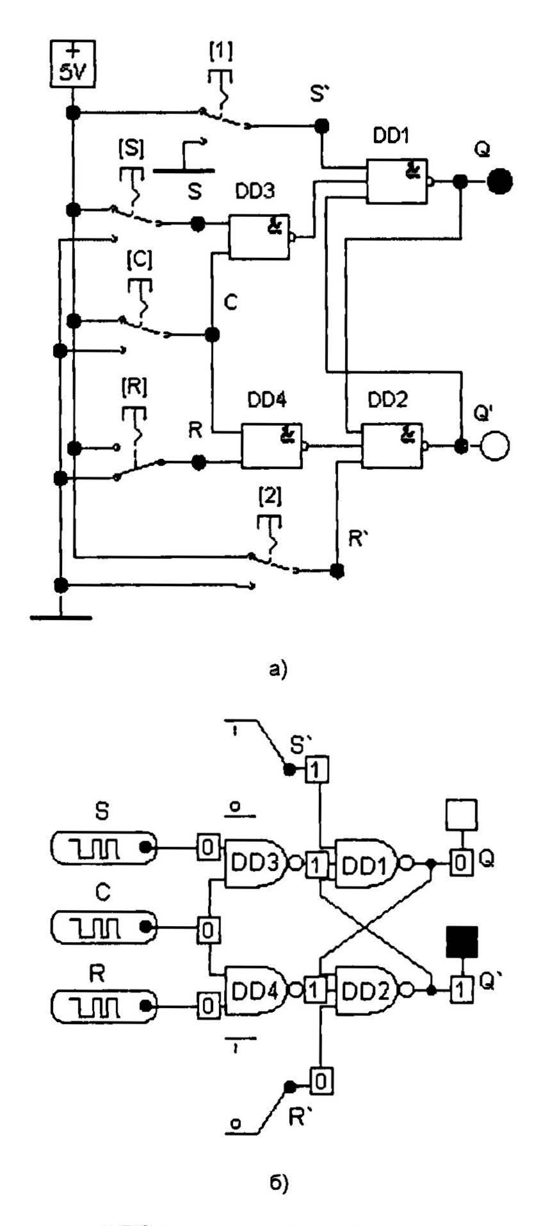


Рис 126 Синхронный RS-триггер с предустановочными входами

Двухступенчатый синхронный RS-триггер

Рассмотренные до сих пор триггеры имели одну ячейку памяти (одноступенчатый триггер) В двухступенчатых триггерах, состоящих из двух ячеек, первая ячейка (ведущая) является входной, а вторая (ведомая) – выходной. Поэтому такую структуру называют MS-структурой (от англ. Master – Slave, означающего хозяин – слуга).

Для построения двухступенчатого синхронного RS-триггера можно снять копии с одноступенчатых триггеров и соединить их необходимым образом. В программе **EWB** вновь используем для этого способ создания функциональных субблоков.

Вернемся к схеме RS-триггера, показанной на рис. 126,а. Путем простого растяжения выделенных частей придадим ей такой вид, чтобы окончательно внутрь выделяющей рамки входил только один триггер (см рис. 127,а)

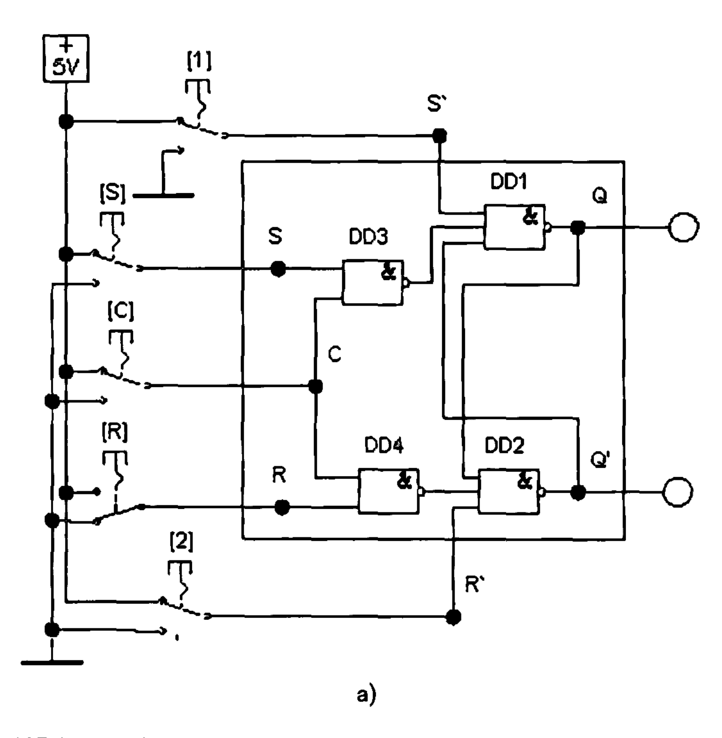


Рис. 127 (начало)

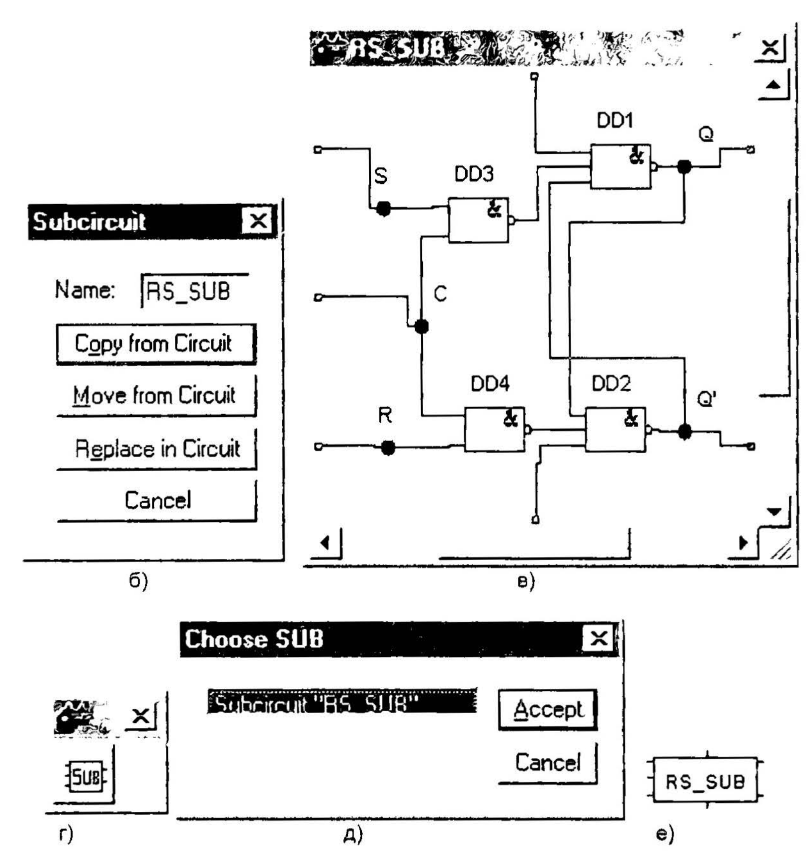


Рис 127 (окончание) Создание субблока RS-триггера (**EWB**)

После выделения субблока, пока его элементы имеют активный красный цвет, входим в меню Circuit (цепь) и выбираем Create Subcircuit (создание субцепи). В появившемся меню Subcircuit (см рис 127,б) впечатываем в строке Name название, например, RS_SUB и выбираем Copy from Circuit (копирование из цепи) На рабочем поле появится дополнительное окно с развернутой схемой субблока (рис 127,в).

Для выбора схемного изображения субблока в свернутом виде, надо нажать в ряду выбора компонентов на иконку Favorites (избранные) . В результате получим иконку с изображением

субблоков (см рис. 127,г) и возможность дальнейшего выбора Choose SUB (см. рис. 127 д). Нажав в последнем окне ЛКМ Accept (согласиться), получим искомое схемное изображение субблока в свернутом виде (рис. 127,е)

Скопируем теперь схемное изображение субблока в свернутом виде (рис 127,е) в буфер обмена. Откроем новое окно (New), нажав на иконку Favorites , увидим, что внутри нет никаких элементов Нажмем ЛКМ Past: на экране возникнет схемное изображение субблока, а в открытом окошке Favorites иконка для его выбора (как ранее на рис 127,г) Скопировав еще один схемный субблок (или выбрав его из раздела Favorites), соберем схему двухступенчатого RS-триггера со структурой MS (см. рис. 128,а) Здесь тактирование ведомой ступени (Slave) выполняется антисинхронно по отношению к первой (Master) за счет подачи тактового импульса через инвертор Отсюда другое название подобного триггера – двухтактный

Воспользовавшись описанной выше процедурой создания субблоков, создадим на основе схемы по рис. 128,а субблок двухтактного триггера (см рис. 128,б,в). Название этого триггера RSFF означает, что это RS Flip-Flop (щелчок – хлопок) триггер Соберем схему для исследования работы этого триггера (см рис 128,г). На предустановочные асинхронные входы подадим неактивные логические уровни (S`=R`=1). Пусть вначале C=0, S=1, R=0. Включим моделирование: на выходах установится неустойчивое состояние. Переведем C=1 и затем C=0: получим Q=1 и Q`=0. Переведем S=0, R=1, потом C=1 и затем C=0 получим Q=0 и Q`=1. Наблюдая за работой этого триггера, видим, что инфор-

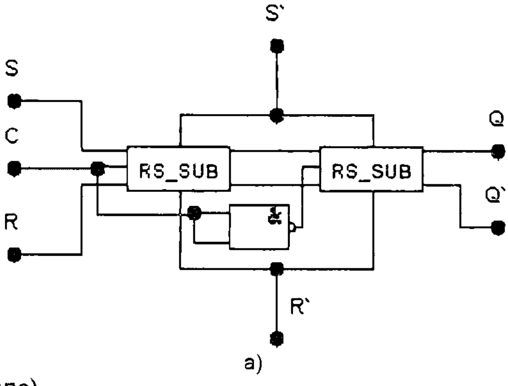


Рис 128 (начало)

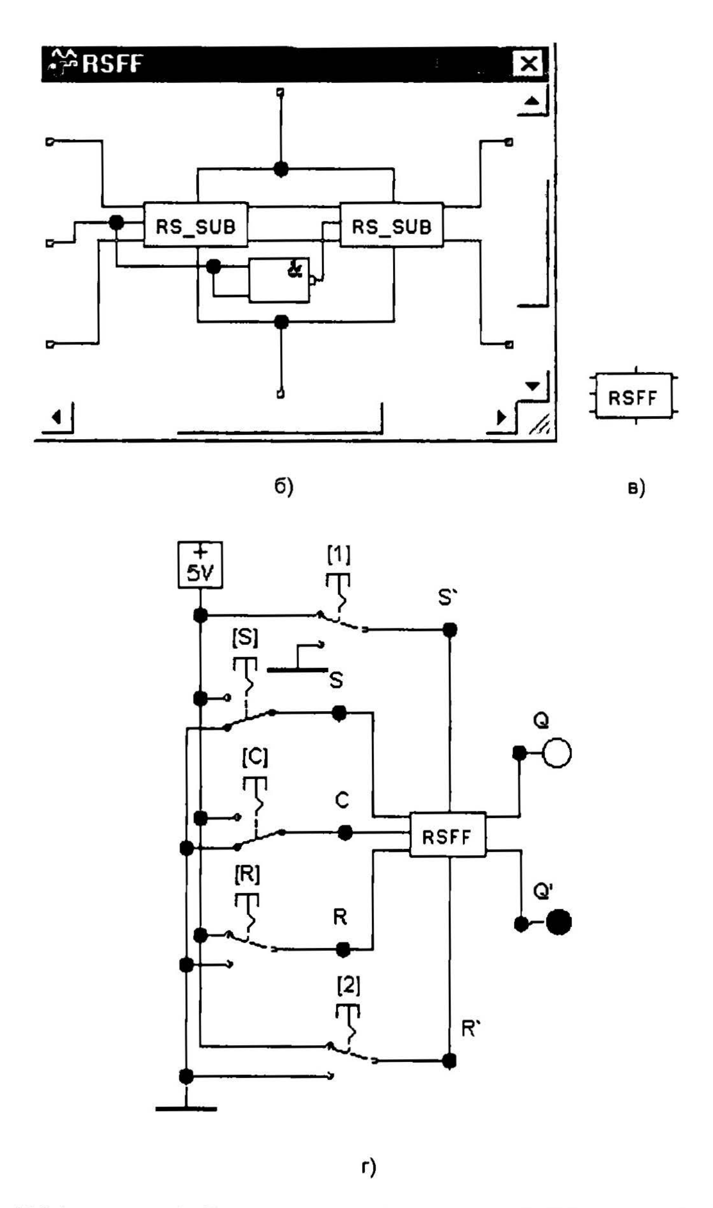


Рис 128 (окончание) Двухступенчатый синхронный RS-триггер (**EWB**)

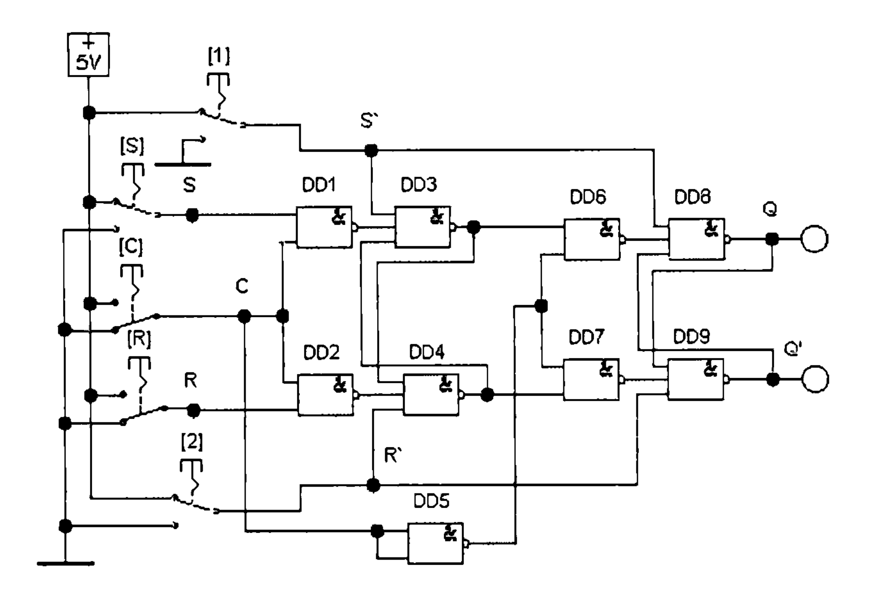


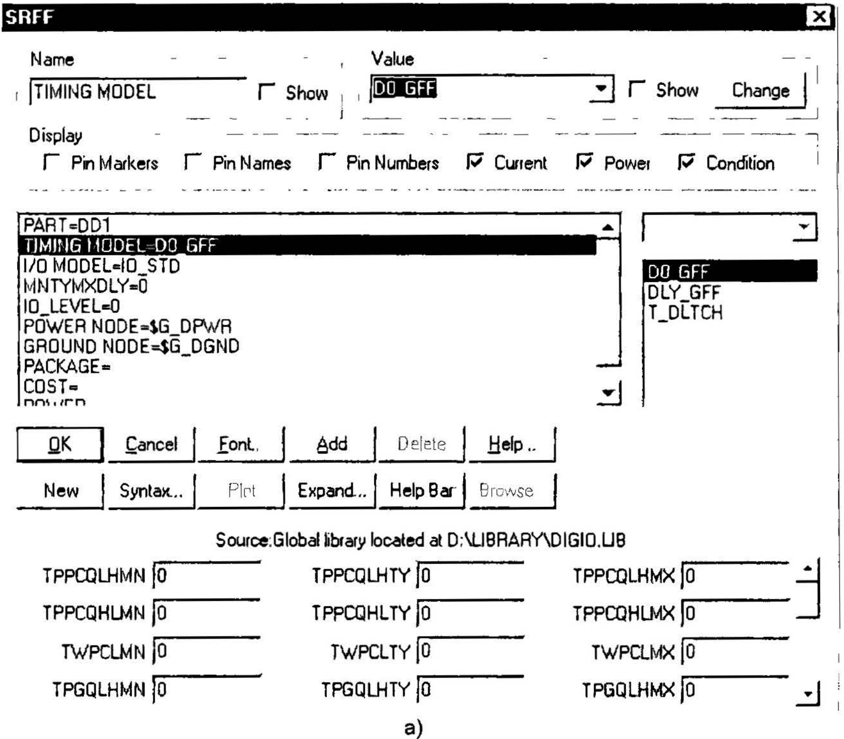
Рис. 129. Логическая структура RSFF-триггера (**EWB**)

мация заносится в первую ступень на фронте синхроимпульса (как бы первый такт – Flip), а затем после внутренней задержки, обусловленной блокировкой инвертором второй ступени, при окончании импульса во вторую ступень (как бы второй такт – Flop). В заключение приведем полную логическую структуру триггера RSFF (см. рис 129), которая может быть получена как прямой поэлементной сборкой, так и обратной реконструкцией из субблоков RSFF и RS_SUB. Сопоставление схем на рис 128,г и 129 демонстрирует эффективность использования «техники субблоков» для создания виртуальных моделей. Эта эффективность будет расти по мере роста сложности схем и повторяемости типовых субблоков.

В программе **МС** имеется соответствующий готовый компонент, обозначенный как SRFF. Для его выбора подадим команды: Component>Digital Primitives>Gated Flip-Flop/Latches>SRFF. На рабочем поле возникнет УГО триггера и окно для задания его атрибутов и характеристик (рис. 130,а). Впечатываем в окошке PART позиционное обозначение, например DD1, и выбираем временную модель (TIMING MODEL) идеального триггера D0_GFF, оставляя прочие характеристики неизменными. Подтверждаем сделанный выбор (OK) и получаем на рабочем поле

соответствующий схемный компонент, в котором отсутствует разметка выводов, за исключением указания их инверсности (см. рис 130,б) Однако, разметка выводов совпадает с той, которую мы использовали ранее в соответствующем субблоке программы **EWB** (см рис 128,г). Поэтому, войдя в раздел Component>Analog Primitives>Miscellaneous>Bubble1, выбираем контактный узел и, задав ему необходимую текстовую метку в появившемся окне, размещаем ее соответственно назначению на УГО (см. рис. 130,б).

Для исследования работы этого триггера соберем схему, показанную на рис. 130,г, используя источники сигналов из схемы на рис. 125,в. На предустановочных входах выставим нейтральные уровни S`=R`=1. В результате анализа получаются графики (см. рис 130,д), мало отличающиеся, при выбранных параметрах, от показанных на рис 125,г, где на начальном участке вплоть до 0,3 мкс видна некорректная работа (из-за отсутствия предустановки).



а)

Рис. 130 (начало)

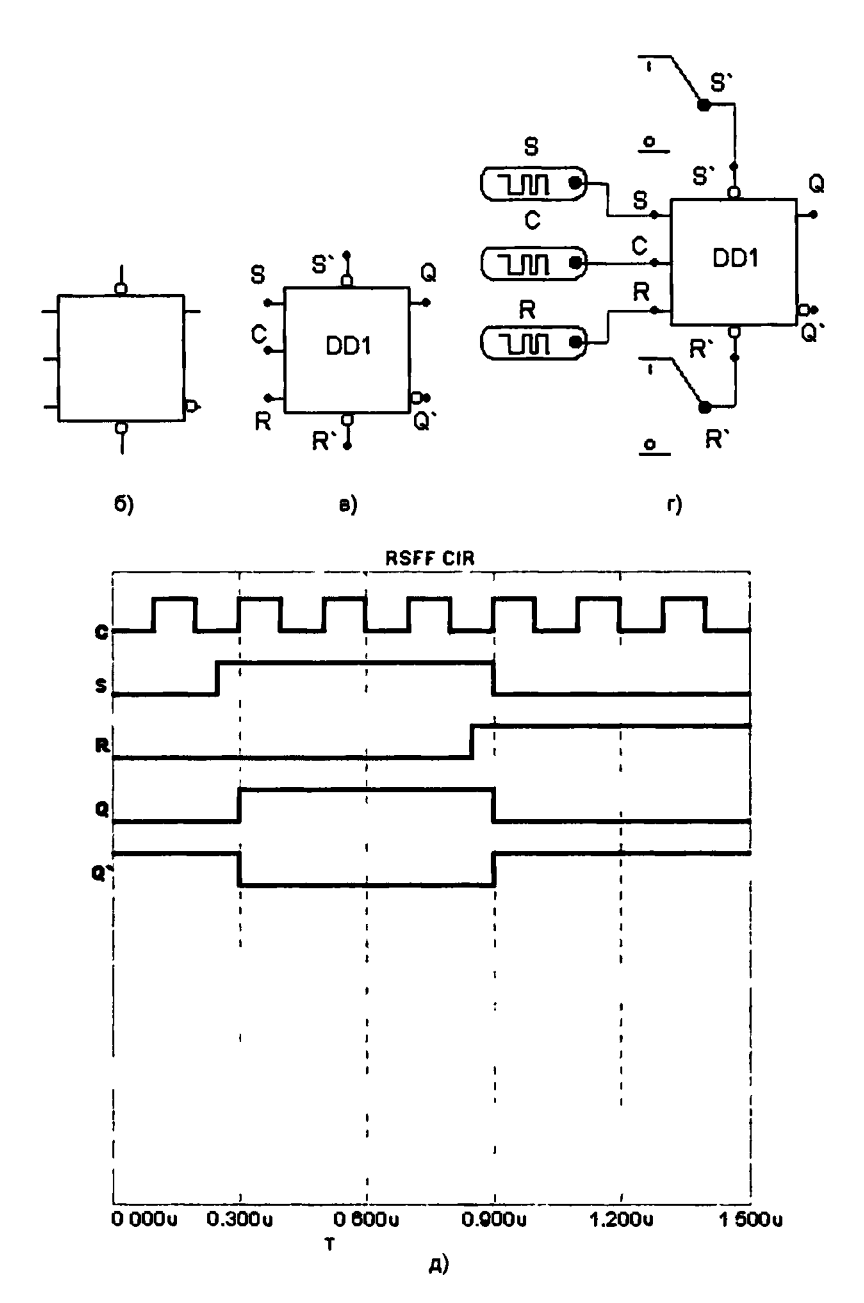
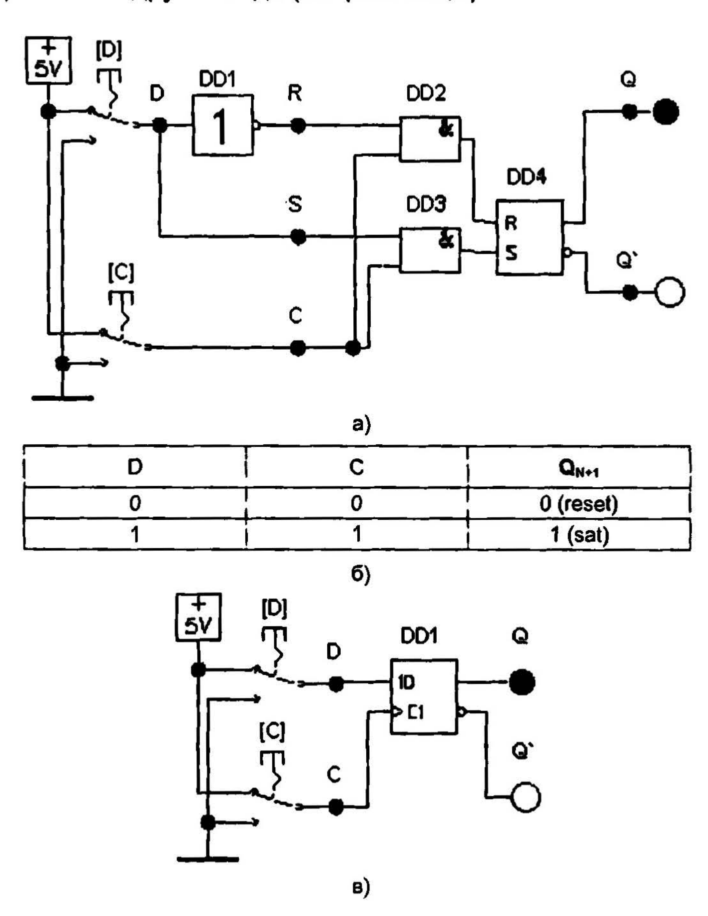


Рис 130 (окончание) Моделирование RSFF-триггера (MC)

Серьезным недостатком RS-триггера является наличие в нем неопределенных состояний. Этого можно избежать различными способами. В так называемом D-триггере (см рис 131,а), входы R и S объединены через инвертор DD1 в один информационный вход данных D (от Date – данные или Delay – задержка) Данный триггер имеет статическое управление и два устойчивых состояния. Иногда этот тип триггера называют прозрачной защелкой, поскольку выход Q повторяет входной сигнал D, пока C=1, и запоминает логический уровень входного сигнала при C=0 Образовав из ЛЭ DD2–DD4 субблок RS_C, эту схему можно представить в другом виде (см рис. 131,б)



а)

D	C	Q_{N+1}
0	0	0 (reset)
1	1	1 (sat)

б)

в)

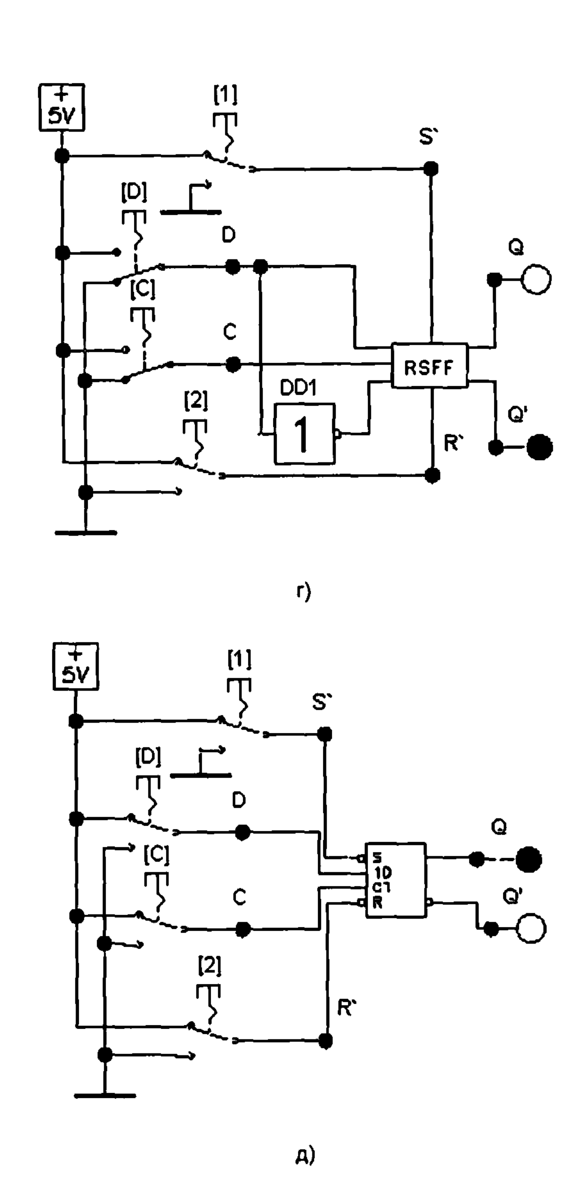
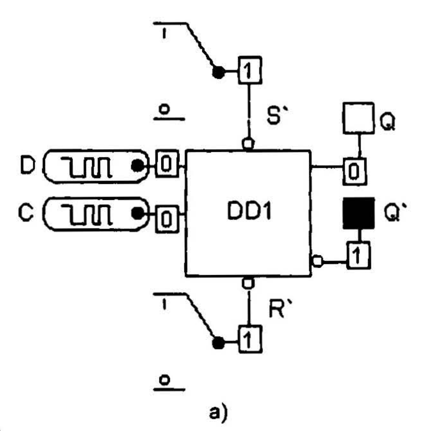


Рис 131 D-триггеры (EWB)

В библиотеке моделей программы **EWB** имеются два D-триггера. Первый представляет собой D-триггер без предустановочных входов и с динамическим управлением (см. рис. 131,б) Последнее в УГО триггера отмечено треугольным концом стрелки у тактового входа. Эта помета означает, что триггер срабатывает на положительном фронте тактового импульса; обратное направление означало бы срабатывание на срезе импульса. В литературе часто используют более лаконичные пометы в виде косой черточки на этом входе, наклоненной вправо для фронта или влево для среза (подобно боковым сторонам воображаемого трапецеидального или треугольного импульса) В программах **EWB** и **MC** синхронизация на срезе импульса отмечается дополнительным кружком около стрелки тактирующего входа Триггерная таблица в сжатом виде показана на рис. 98,в. В ней Q_{N+1} – значение выхода на такте N+1 Воспользовавшись ранев созданным субблоком RSFF или библиотечной моделью D-триггера, можно также собрать схемы для исследования работы триггеров с предустановочными входами (см рис. 131,г,д)

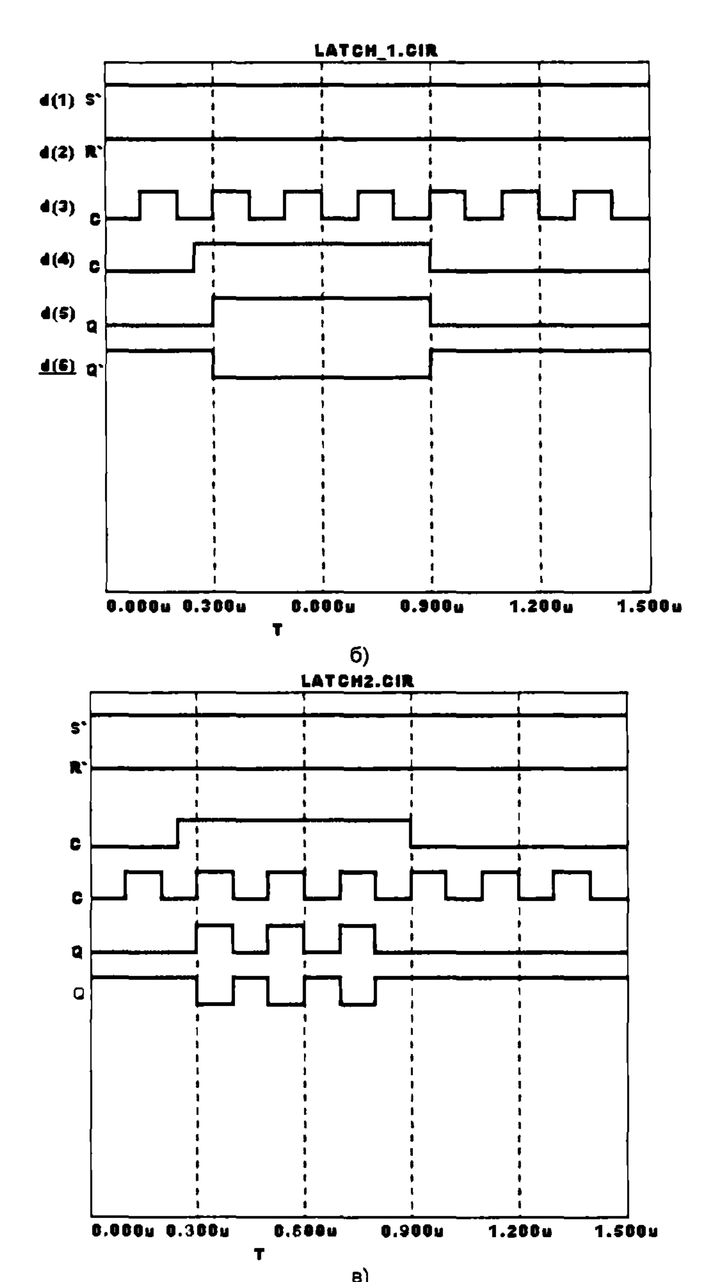
В программе **MC** для исследования работы D-триггера возьмем библиотечную модель LATCH (англ. – защелка), представляющую собой однотактный синхронный D-триггер с асинхронными инверсными установочными входами R\`и S\` (см рис 132,а)

Воспользуемся также тактовым сигналом С со схемы рис 126,б и с нее же возьмем сигнал S, используя его как D Проведя анализ, получим графики, показанные на рис. 132,б Поменяв сигналы С и D местами, получим графики, показанные на рис. 99,в



а)

Рис. 132 (начало)



б)

в)

Рис 132 (окончание) D-триггеры (MC)

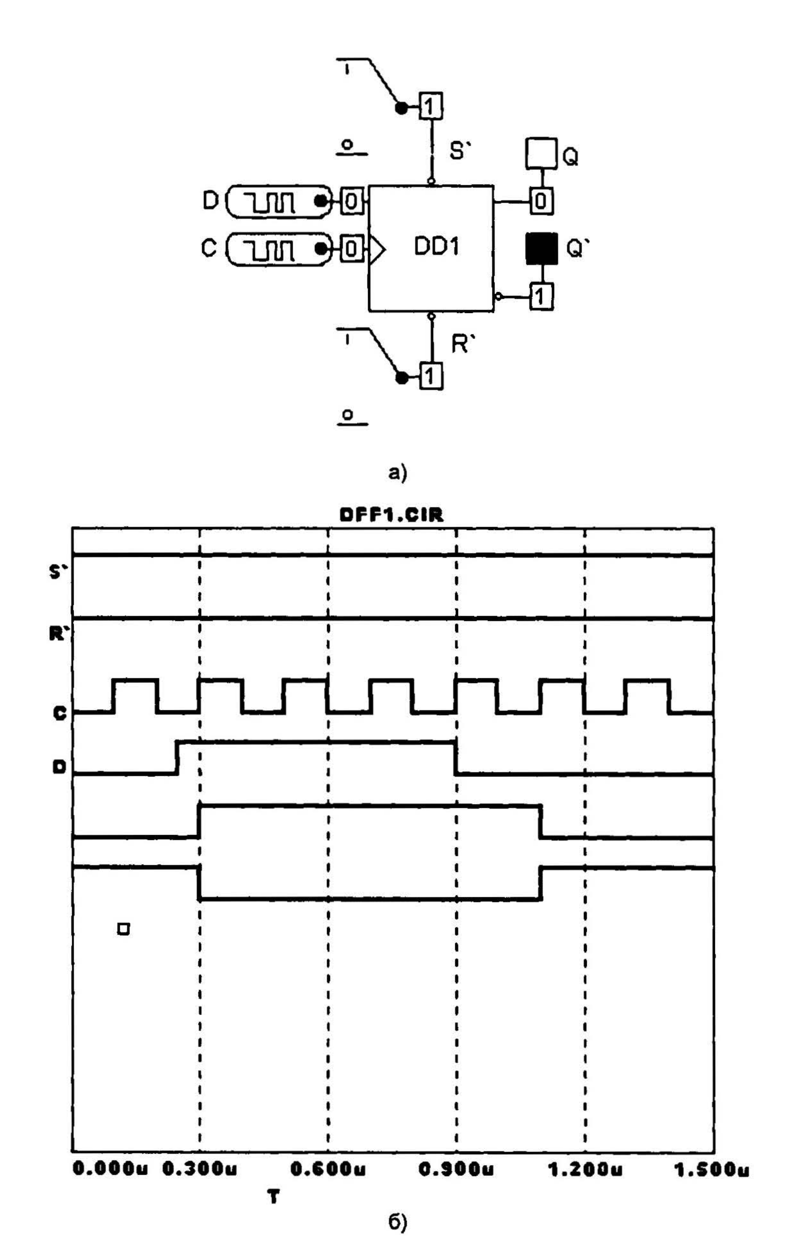


Рис 133 DFF-триггер (**MC**)

Заменив в схеме на рис 132,а триггер LATCH на библиотечную модель триггера DFF, выбираемую командами: Component>Digital Primitives>Edged-Trggered Flip-Flops (триггеры управляемые фронтом)>DFF, получим возможность исследования триггера с динамическим управлением (см рис. 133,а). Используя те же источники сигналов, получим соответствующие графики, показанные на рис. 133,б.

Из приведенных графиков видны особенности работы D- и DFF-триггеров В последнем случае информация заносится в память по положительному фронту тактового импульса и сохраняется там до следующего импульса

Т-триггер

На основе двух синхронных RS-триггеров можно собрать схему так называемого счетного или Т-триггера (от англ. Toggle – чека, кривошип), имеющего один вход (см. рис. 134,а).

Текущее состояние Т-триггера определяется его же состоянием в текущем такте. D-триггер, показанный на рис. 133,а, также может быть преобразован в Т-триггер (см. рис. 134,в). Счетный режим в D-триггере организуется за счет обратной связи: сигнал, подающийся с инверсного выхода триггера на его вход D, приводит к смене его состояний. Из графиков (см. рис. 134,г) видно, что

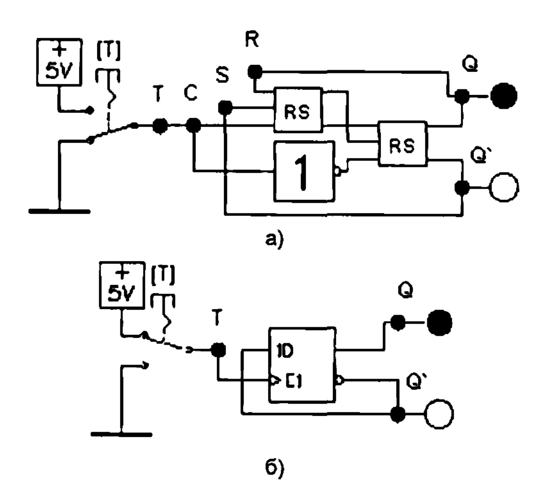


Рис 134 (начало)

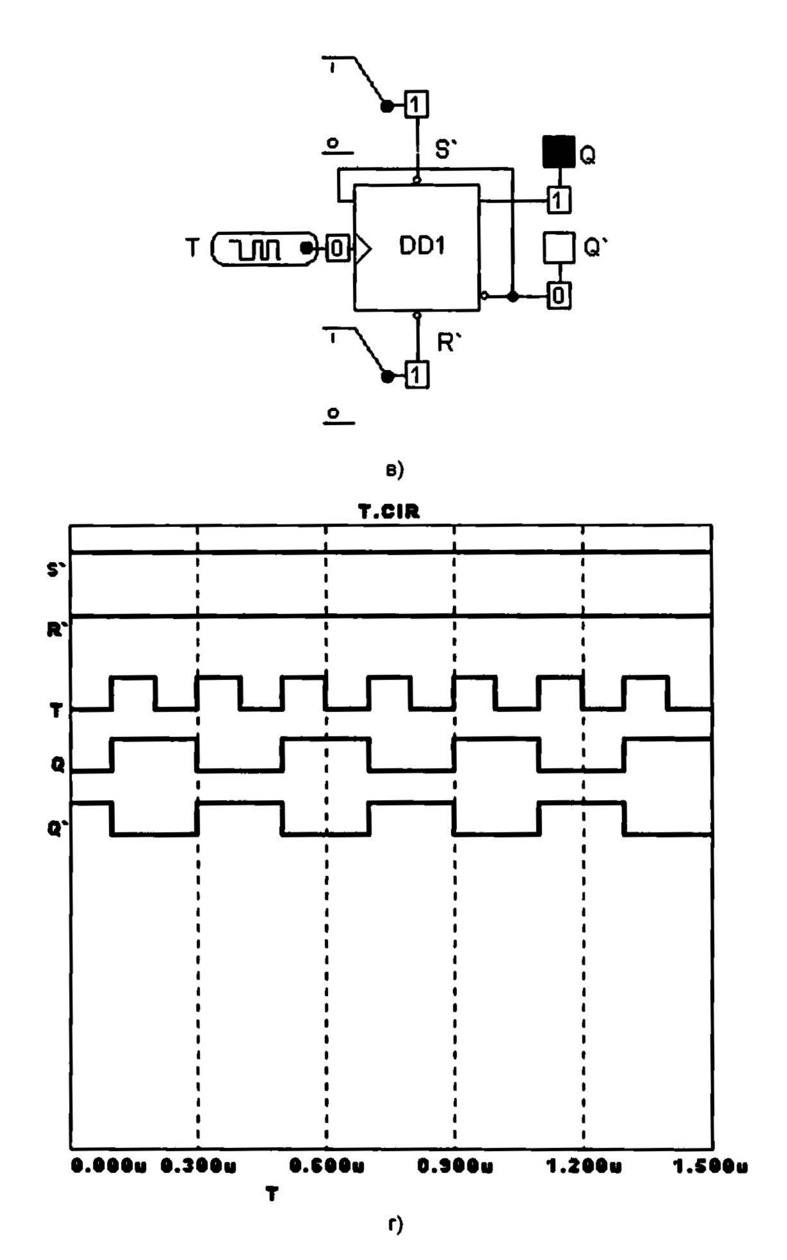
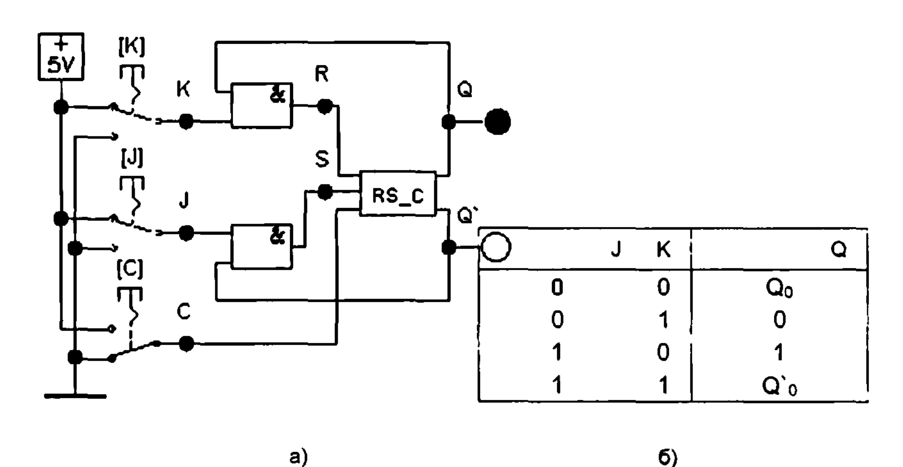


Рис 134 (окончание) Т-триггеры

смена состояний здесь происходит всякий раз, когда входной сигнал изменяет свое значение в одном направлении. В данном случае смена состояний происходит на каждом фронте (положительном перепаде напряжения) текущего импульса Т-триггер, изменивший свое состояние на фронте первого импульса (с 0 на 1), находится в режиме хранения (1) до прихода фронта второго импульса, который вновь меняет его состояние (с 1 на 0), наступает режим хранения (0) и т д Поскольку длительность нахождения в каждом состоянии оказывается равной периоду следования тактовых импульсов, то частота смены состояний триггера оказывается в два раза меньшей, чем частота следования тактов В результате происходит деление частоты на два: частота следования выходных импульсов равна половине частоты следования входных импульсов

Универсальный JK-триггер

Как уже отмечалось выше, основным недостатком RS-триггеров является неопределенность запоминаемого состояния при S=R=1 В D- и Т-триггерах эта неопределенность снята, но и их функциональные возможности уменьшились JK-триггер является, пожалуй, наиболее удачным усовершенствованием RS-триггера Дополняя RS-триггер определенной логической структурой управления, можно создать условия, при которых выход триггера будет всегда наперед известен при любых сочетаниях входных сигналов



J	K	Q
0	0	Q_0
0	1	0
1	0	1
1	1	$Q`_0$

а) б)

Рис 135 (начало)

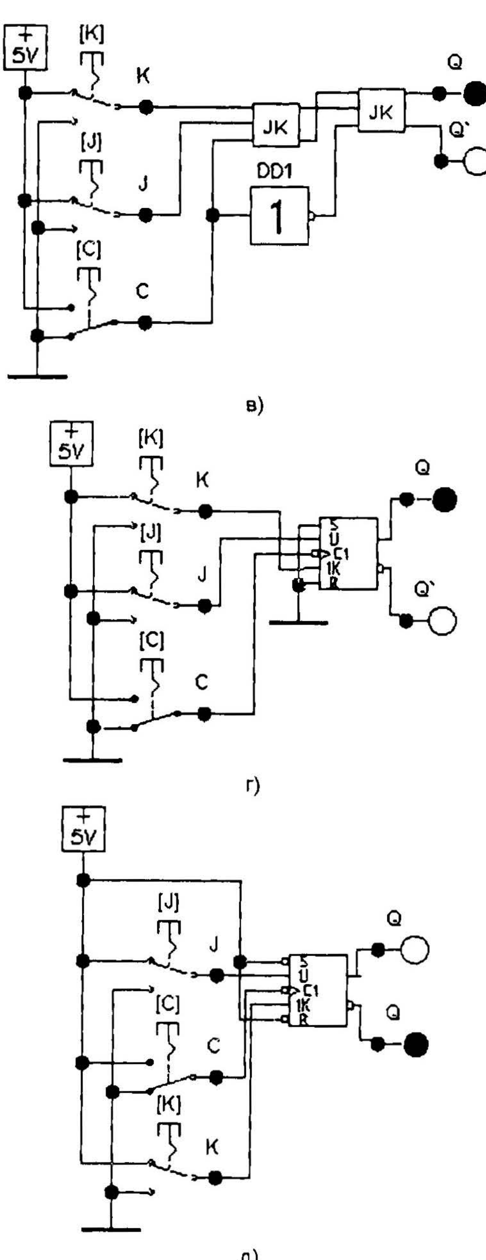


Рис 135 (окончание) JK-триггеры (**EWB**)

В схеме на рис 135,а синхронный RS-триггер дополнен входной логикой на двух ЛЭ AND так, что образуются обратные связи между противоположными входами и выходами: R–Q и S–Q` В результате появляются два новых входа. сброса – K (от англ. – Killer – убийца) и установки – J (от англ. Jerk – толчок, резкое движение). Эти входы синхронизируются тактирующим входом C. Задавая различные значения входам J и K в этой схеме и наблюдая за состоянием выхода Q, можно составить триггерную таблицу состояний (см рис. 135,б). Здесь Q_0 и Q_0` состояние выхода в предыдущем такте. Клавишу C надо нажимать на короткий промежуток времени, выполняя C=1, и возвращать обратно в C=0 Образовав субблок JK из этой схемы, можно составить двухступенчатый JK-триггер по типу M–S (см рис 135,в) Наконец, войдя в раздел триггеров библиотеки компонентов программы **EWB**, можно составить еще две схемы для испытаний JK-триггеров с предустановками и динамическим управлением (см рис 135,г,д) Здесь кружком около тактирующего входа отмечена синхронизация на срезе импульса.

В программе **MC** JK-триггер выбирается командами· Component>Digital Primitives>Edged-Triggered Flip-Flops (триггеры, управляемые фронтом)>JKFF. С этим компонентом построим схему, показанную на рис. 136,а. Используя те же источники сигналов, получим соответствующие графики, показанные на рис 136,б

Совет (**MC**) Не забывайте устанавливать виртуальные ключи асинхронной предустановки триггеров перед началом моделирования в неактивное положение (S`=R`=1).

JK-триггер называется универсальным не только потому, что он отрабатывает все возможные сочетания входных сигналов, но и потому, что на его основе можно реализовать другие типы триггеров Так, объединив все входы, кроме тактового, в один и подав на них лог 1, получим счетный, или T-триггер (см. схему в программе **EWB** на рис 137,а) Для более детального исследования работы этого триггера подадим на его вход синхроимпульсы от функционального генератора (Function Generator), а состояние входов и выходов будем контролировать специальным логическим анализатором (Logic Analyzer), отмеченным

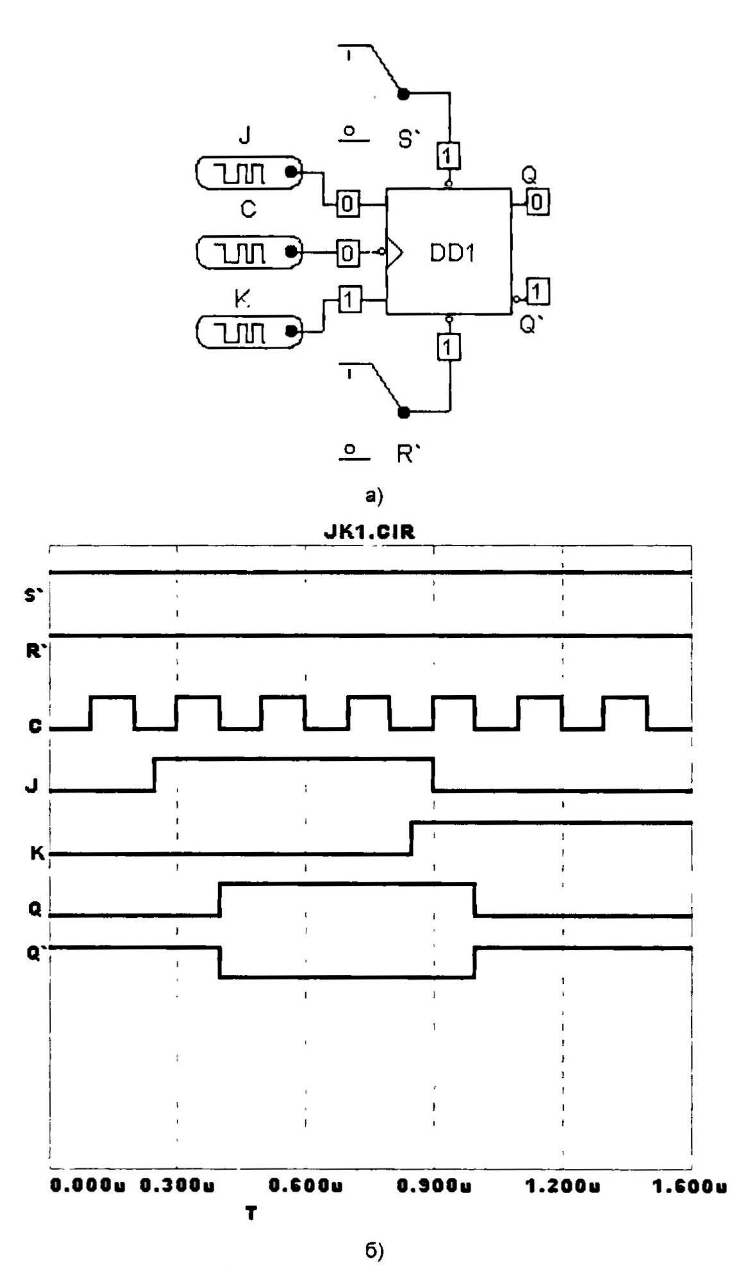


Рис. 136. JK-триггер (**MC**)

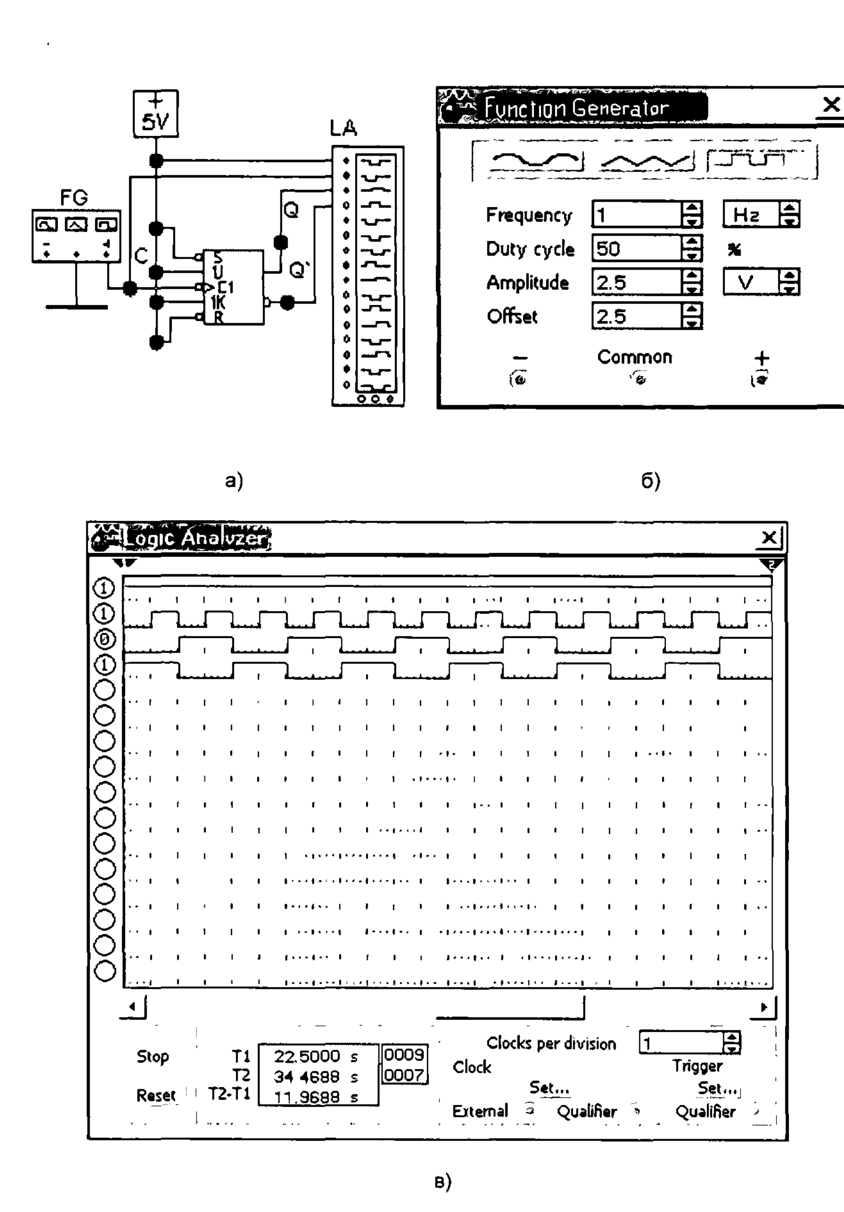
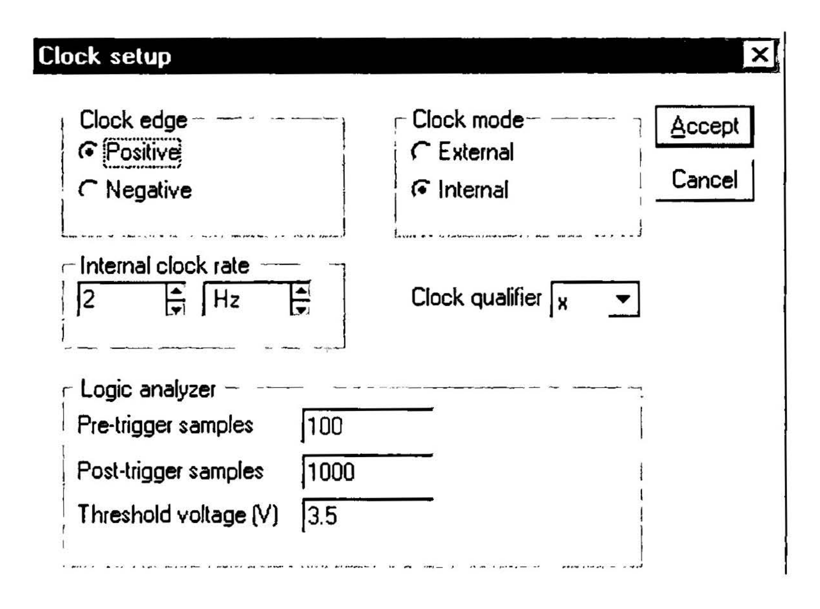
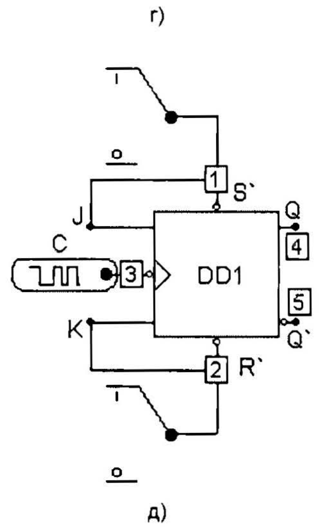


Рис. 137 (начало)



г)



д)

Рис 137 (продолжение)

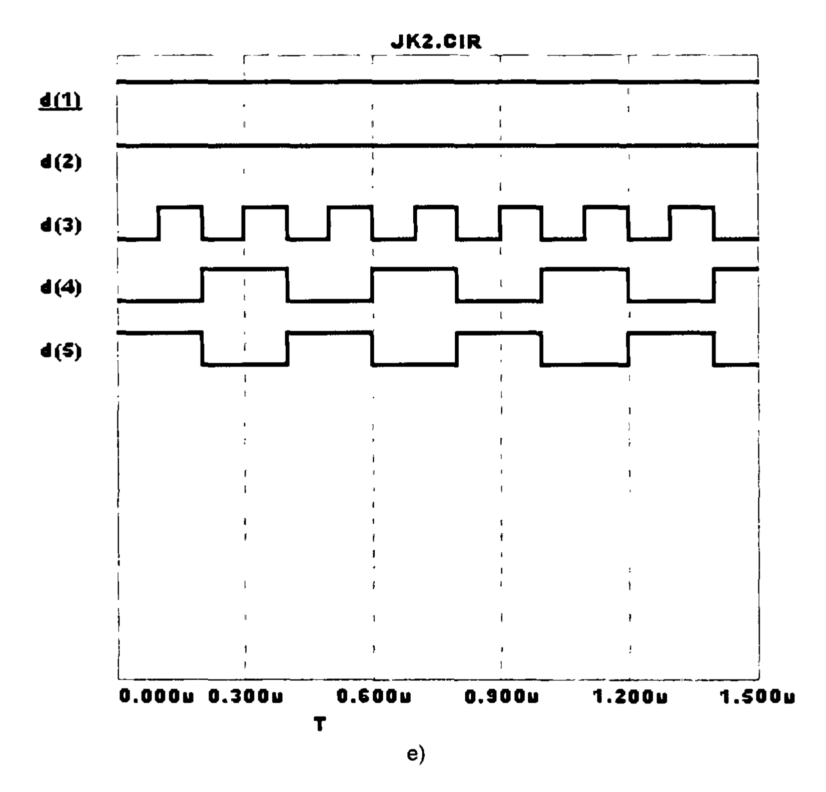


Рис. 137 (окончание). Т-триггер на основе JK-триггера

метками FG и LA, соответственно, на рис. 137,а. Эти виртуальные приборы находятся в панели Instruments (инструменты).

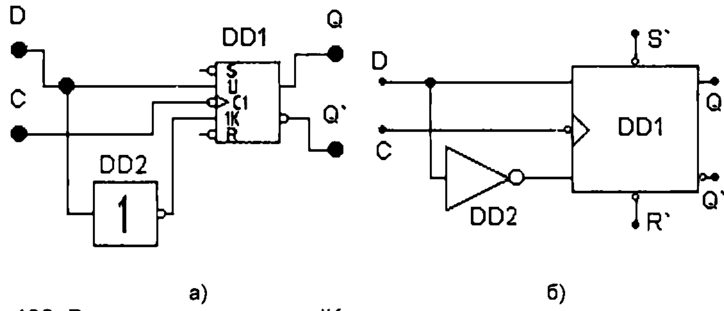
Дадим некоторые пояснения к показанной схеме. На асинхронные установочные входы подан высокий (неактивный) уровень напряжения S\`=R\`=1. На информационные входы подана логическая единица, т.е. J=K=1. К входу С подключен генератор тактовых импульсов. На схемном УГО генератора FG имеются выходные клеммы «+», «Common» (общая) и «–». Использованы клеммы: «+», соединенный с С и «Common», которая заземлена. Эти же клеммы видны на виртуальной панели (см. рис. 137,б), но здесь они «не работают» и не надо пытаться к ним подключаться.

Эта лицевая панель служит для задания режима генератора. Выбираем последовательность прямоугольных импульсов с частотой следования (Frequency) 1 Hz, коэффициентом заполнения (Duty cycle) 50%, амплитудой (Amplitude) 2.5 V и смещением (Offset) также 2 5 V (это дает положительные униполярные импульсы с максимальным значением 5 V).

На схемном изображении логического анализатора LA вертикальный ряд клемм служит для подключения исследуемых цифровых сигналов. Развернув двойным щелчком ЛКМ по схемному изображению LA откроем его лицевую панель (рис. 137,в). На ней нажмем кнопку Set установки режима развертки логического анализатора на открывшейся панели Clock setup (см рис 137, г) Здесь выбираем режим внутренней развертки (Internal clock rate) 2 Hz С остальными параметрами, заданными по умолчанию, соглашаемся. Включив моделирование, получим следующие графики цифровых сигналов на рис. 137,в сверху вниз: лог. 1 на входах S`, J, K, R; чередующиеся с частотой 1 Гц лог 1 и лог 0 на входе C и аналогичные последовательности на выходах Q и Q`, но с вдвое меньшей частотой, т е 0 5 Гц Подобное деление частоты на 2 произошло из-за того, что триггер перебрасывается в новое состояние на каждом спаде тактовых импульсов, что хорошо заметно на графиках. Это поведение можно было бы предсказать, если обратиться к последней строке таблицы рис 135,б Здесь видим, что при J=K=1 прямой выход триггера будет повторять значение инверсного выхода в предыдущем такте Поскольку синхронизирующий вход здесь динамический и инверсный, то переключения будут происходить на срезе (спаде) каждого тактового импульса.

Мы упорно не пользуемся весьма распространенным термином «задний фронт», считая его безграмотным Ведь само слово «фронт» происходит из нем Front от франц. Front, буквально означающего лоб. «Задний фронт», конечно, похож на оксюморон типа «горячий снег», но если чуть-чуть вдуматься, то звучит ужасно. Все-таки лоб у человека думающего находится впереди. Конечно, несколько лучше звучит «отрицательный фронт», но какой-то осадок остается.

Соберем далее схему, аналогичную рис. 137,а, пользуясь средствами программы **MC** (см. рис. 137,д). На полученных графиках (рис. 137,е) нумерация d1–d5 соответствует нумерации цифровых узлов на схеме рис. 137,д. Здесь также хорошо видно деление тактовой частоты на 2. Схемы D-триггеров на основе JK-триггера показаны на рис. 138,а,б.



а) б)
Рис. 138. D-триггеры на основе JK-триггера

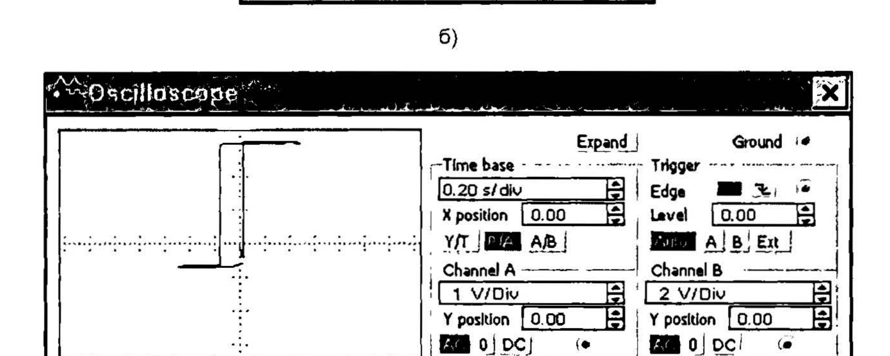
Триггер Шмитта

Рассмотренные выше разновидности триггеров относились к группе симметричных, но наряду с последними в цифровых устройствах используются также и несимметричные триггеры Один из типов подобных триггеров традиционно называют триггером Шмитта по фамилии О. Г. Шмитта, который в 1938 г. предложил соответствующую ламповую схему на двойном триоде, охваченном слабой положительной обратной связью. Иногда встречается и иное написание названия: «Триггер Шмидта», хотя в англоязычной литературе соответствующее устройство называется «Schmitt trigger».

Основу несимметричных триггеров, как и симметричных, составляет двухкаскадный усилитель постоянного тока, охваченный положительной обратной связью. Однако каскады здесь уже не идентичны по своим параметрам и связям между ними, но главное отличие заключается в том, что выходной сигнал в отсутствие входного однозначно определен. Поэтому подобные триггеры не обладают памятью и используются как спусковые устройства либо для формирования последовательности прямоугольных импульсов из сигналов произвольной формы, например синусоидальных. Вообще данный тип триггеров ближе к импульсным, нежели к цифровым устройствам. При классификации цифровых микросхем они и вовсе попадают «между двух стульев» и их называют: «Логические элементы – триггеры Шмитта» Выпадают они и из нашего контекста, так как для рассмотрения их работы придется вернуться к элементной базе низкого уровня: транзисторам

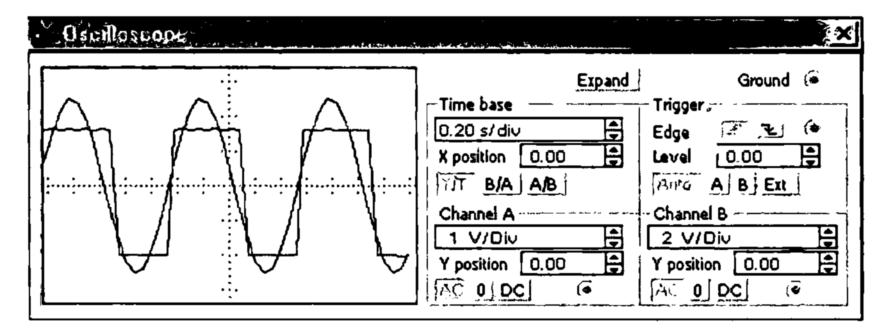
а)

б)

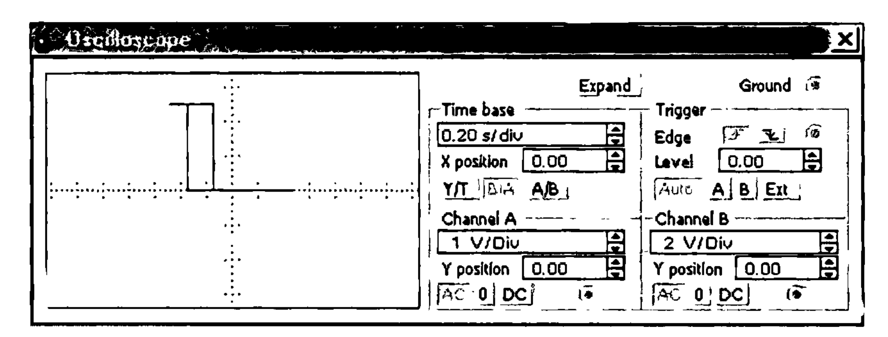


в)

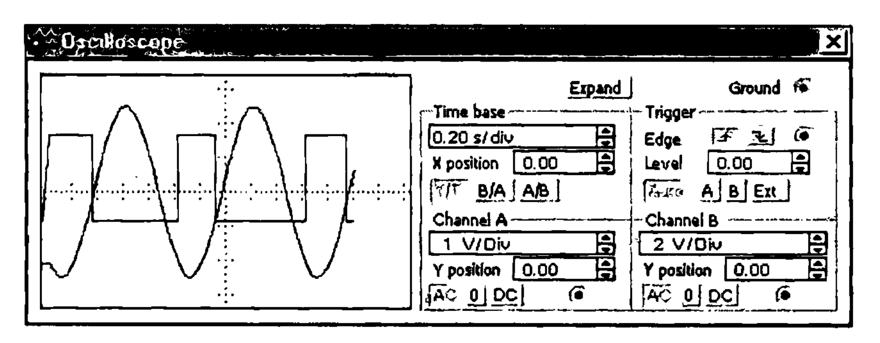
Рис 139 (начало)



г)

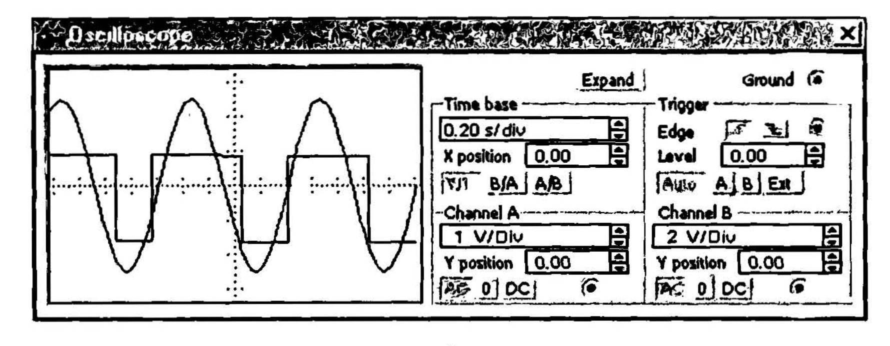


д)



е)

Рис. 139 (продолжение)



ж)

Рис. 139 (окончание). Триггер Шмитта (**EWB**)

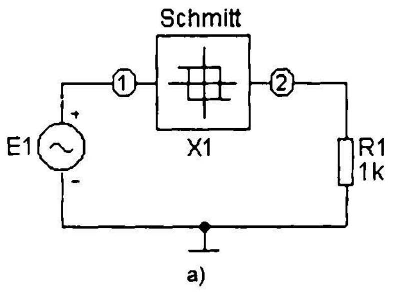
Рассмотрим в программе **EWB** работу классической схемы триггера Шмитта на двух транзисторах (VT1 и VT2) с эмиттерными связями (см. рис. 139,а). Входной сигнал от функционального генератора FG подается на вход In (база VT1) и канал А осциллоскопа OSC, а выходной снимается с вывода Out (коллектор VT2) и подается на канал В. Для снятия передаточной характеристики триггера выставим на генераторе режим сигналов треугольной формы с параметрами, показанными на рис. 139,б. Для того чтобы получить зависимость выходного напряжения от входного, на осциллоскопе выберем режим развертки типа В/А (см рис 139,в) Поскольку далее, для сравнения, будет выполняться моделирование триггера Шмитта на типовых БЛЭ (DD1 и DD2), то схема предусматривает коммутацию приборов ключами [Space] и [С]. В данном же случае ключи [Space] должны находиться в верхнем положении, а ключ [С] – в любом.

Включив моделирование, получим на экране характерную петлю гистерезиса (рис. 139,в). Термин «гистерезис» был введен в научный обиход в 1900 г. английским ученым Д.А Юингом (Ewing J.A.) применительно к намагничиванию ферромагнетиков. Дословным переводом этого термина, имеющего греческую основу (hysterësis), является «недостаток», «нехватка». В науке и технике под явлением гистерезиса подразумевают отставание следствия от производящей его причины Именно благодаря магнитному гистерезису «работает» магнитная память (так что побольше бы таких «недостатков»). В зависимости от свойств материалов и вида рассматриваемых явлений наблюдаемая форма гистерезисной зависимости может быть самой разнообразной: от эллиптической до прямоугольной

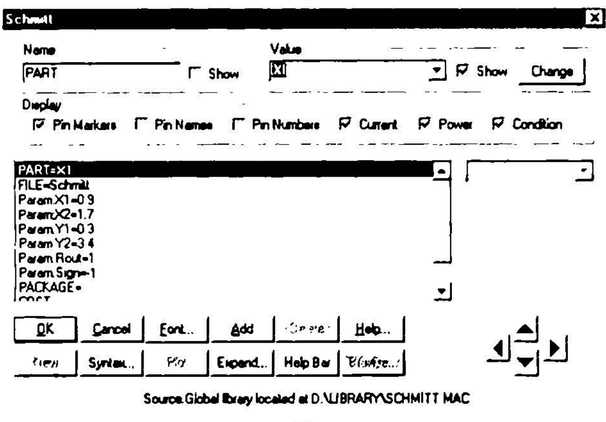
В триггере Шмитта наблюдается отставание величины выходного напряжения от входного. Если частоту следования импульсов уменьшить в десять раз (для этого надо воспользоваться установочными кнопками в окошке Frequency функционального генератора), то можно визуально пронаблюдать, как по мере роста напряжения вычерчивается вся кривая, проходя фигуру против часовой стрелки Такой своеобразный вид передаточной функции триггера обусловлен переключением под действием входного напряжения, регулируемого двумя обратными связями· положительной обратной связью со второго каскада на первый за счет общего резистора R4 и отрицательной обратной связью по току через этот же резистор, когда открыт транзистор VT1 Если теперь переключить генератор на синусоидальные колебания, а осциллоскоп на развертку сигналов во времени (Y/T), то синусоидальные колебания на входе превращаются в синфазные (по основной гармонике) прямоугольные колебания на выходе триггера (см. рис. 139,г), поскольку в данном случае реализован неинвертирующий триггер Шмитта

В комплекте БЛЭ программы **EWB** имеется инвертирующий триггер Шмитта, на УГО которого проставлена «родовая метка» в виде петли гистерезиса (см. компонент DD1 на схеме рис 139,а) Для снятия передаточной характеристики этого триггера надо перевести переключатели [Space] в нижнее, а ключ [C] – в левое положение Установив режим развертки в положение B/A, а генератор на треугольную форму колебаний, получим характеристику, показанную на рис. 139,д. В ней обход петли гистерезиса наблюдается по часовой стрелке. Подав на вход триггера Шмитта DD1 синусоидальные колебания, на его выходе (в точке C) получим противофазные (по основной гармонике) колебания прямоугольной формы (см. рис. 139,е). Наконец, сняв сигнал с инвертора DD2 (переведя ключ [C] в правое положение), вернемся к неинвертирующему триггеру Шмитта (см рис 139,ж). В состав микросхем входят инвертирующие триггеры Шмитта, например в ТТЛ микросхемы 7414 входят шесть подобных инверторов

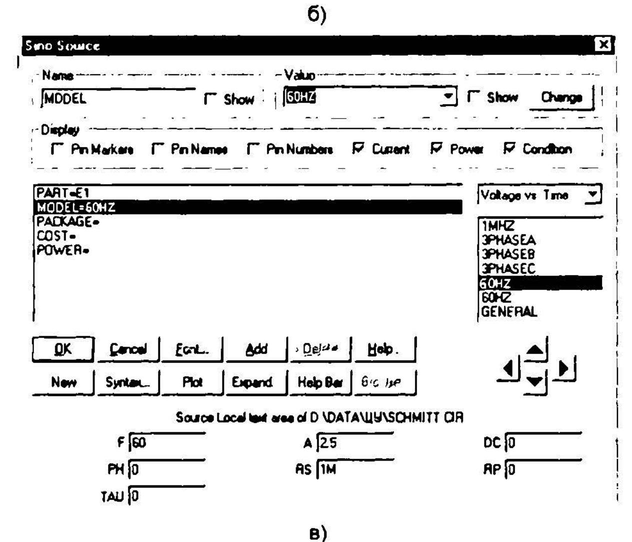
В программе **MC** триггер Шмитта помещен в раздел аналоговых типовых компонентов и выбирается последовательностью команд: Components>Analog Primitives>Macros>Schmitt Поместив выбранный компонент на рабочее поле (см. рис. 140,а), отредактируем его свойства в соответствующем окне (рис. 140,б) Совокупность параметров, описывающих прямоугольную петлю гистерезиса, зададим входными напряжениями (в вольтах) порога



а)

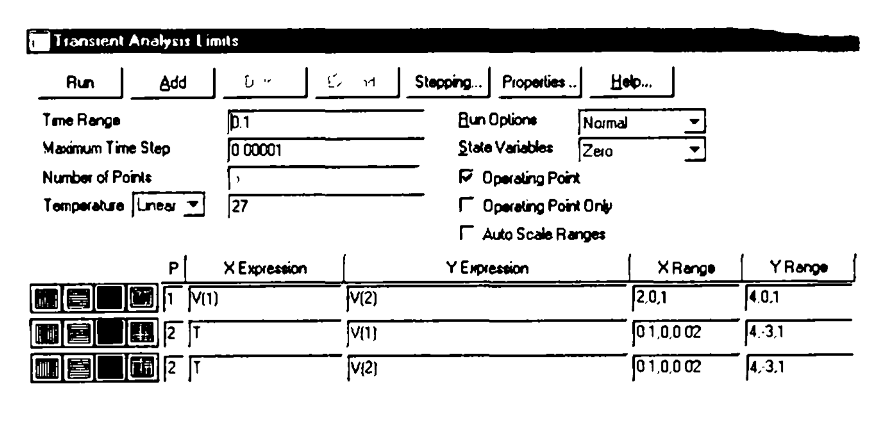


б)



в)

Рис. 140 (начало)



г)

SCHMITT.CIR

4.00
3.00
2.00
1.00
0.00
0.00
1.00
2.00
Umin=0.9V
Uswitch=1.3V
Umax=1.7V
V(2)
V(1)

4.00
3.00
2.00
1.00
0.00
-1.00
-2.00
-3.00
0.00m
20.00m
40.00m
60.00m
80.00m
100.00m
V(1)
V(2)
T

д)

Рис 140 (окончание) Триггер Шмитта (**MC**)

отпускания (Vmin=X1=0.9) и срабатывания (Vmax=X2=1.7) и, соответственно, выходными напряжениями низкого и высокого уровней (Y1=0.3 и Y2=3.4), характерных для распространенных серий ТТЛ микросхем (см. рис. 139,б). Далее выберем в качестве источника входного сигнала генератор синусоидальных колебаний с амплитудой A=2.5 V и частотой 60 Hz (см. рис. 140,в). Собрав схему по рис. 140,в, перейдем к ее анализу. Можно провести анализ на постоянном токе, но более информативным будет анализ переходных процессов (Transient). Установки режимов анализа проводим в соответствующем окне, показанном на рис. 140,г. Здесь первый график представляет передаточную функцию (зависимость V(2) от V(1)), а два других, выводимые в одной координатной системе, отражают временные зависимости этих напряжений. Результаты моделирования показаны на рис 140,д Сопоставляя эти результаты с предыдущими, полученными в программе **EWB** (см. рис. 139,а,б), видим, что это инвертирующий триггер. Нетрудно также видеть, что зона гистерезиса симметрична относительно среднего входного порогового напряжения Uswitch=(1.3 ±0 4) V.

Триггеры Шмитта позволяют эффективно отфильтровать шумы на пологих фронтах сигналов, а также являются незаменимыми для стыковки схем с медленно меняющимися сигналами (<1 Гц) с логическими устройствами типа счетчиков и регистров.

Триггер Шмитта, аналогичный тому, который показан на рис 139,а и выполнен на биполярных транзисторах, можно собрать и на полевых транзисторах В состав микросхем входят инвертирующие триггеры Шмитта, например ТТЛ 7414 содержит шесть подобных триггеров, а микросхема КМОП 4093 (аналог К561ТЛ1) состоит из четырех триггеров Шмитта, на входе каждого из которых стоит двухвходовой элемент *И-НЕ*.

Триггеры Шмитта используются также в аналоговых устройствах при построении разнообразных генераторов импульсов, например треугольной или прямоугольной формы. Простейший подобный генератор состоит из последовательно включенного интегратора и триггера Шмитта. Радиолюбители широко используют триггеры Шмитта в качестве генераторов в самоделках типа сирен, гудков и других звуковых устройств

3.2. Регистры

Вам про чью мебель? Купца первой гильдии Ангелова? Пожа-алуйста. Смотрите на букву А. Буква А, Ак, Ам, Ангелов... Номер? Вот! 82 742.

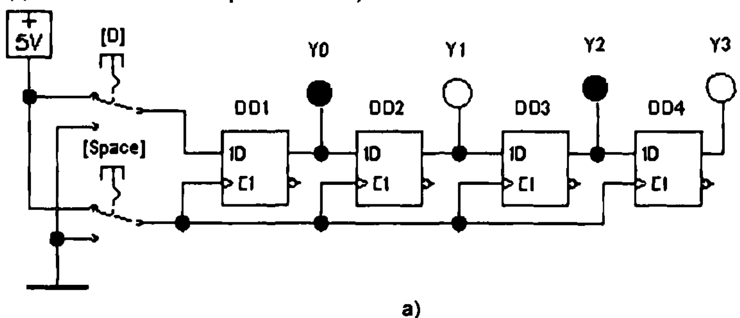
И. Ильф, Е. Петров. Двенадцать стульев

Под регистрами (от лат. registrum – список, указатель) в цифровой электронике и технике подразумевают узлы и устройства, осуществляющие ряд операций с информацией, представленной в виде многоразрядного двоичного кода. К этим операциям относятся: прием, хранение, сдвиг в разрядной сетке, поразрядные логические операции и выдача числовых слов в определенном коде Уже одно перечисление функций регистров показывает, что это самые распространенные узлы цифровых устройств.

Регистры состоят из схем, в которых имеются триггеры и логические элементы По количеству линий передачи переменных регистры делятся на однофазные и парафазные (двухфазные), по системе синхронизации – на однотактные, двухтактные и многотактные. Однако главным классификационным признаком регистров, несомненно, является способ приема и выдачи данных По этому признаку различают параллельные (статические), последовательные (сдвигающие) и параллельно-последовательные регистры.

Последовательные регистры

В последовательных регистрах слова принимаются и выдаются разряд за разрядом. Их называют сдвигающими, так как тактирующие сигналы при вводе и выводе слов перемещают их в разрядной сетке. Сдвигающий регистр может быть нереверсивным (с однонаправленным сдвигом) или реверсивным (с возможностью сдвига в обоих направлениях).



а)

Рис. 141 (начало)

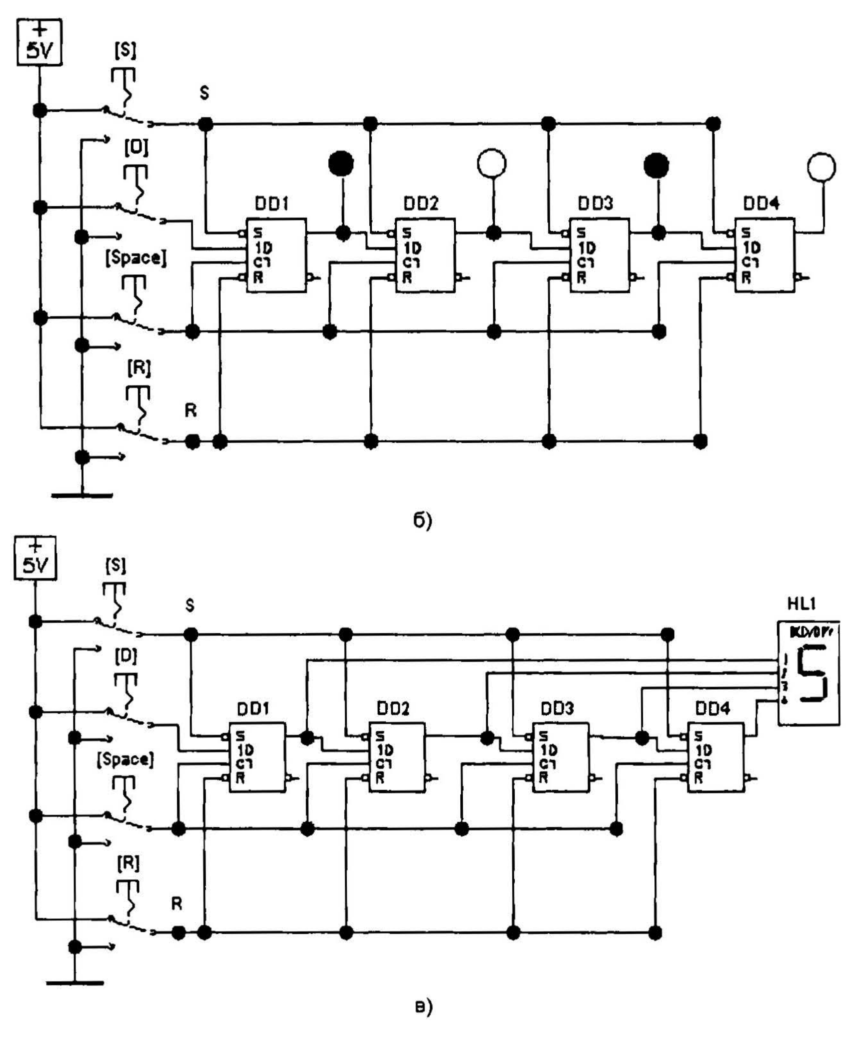


Рис. 141 (окончание). Логическая структура четырехразрядного регистра сдвига (**EWB**)

В программе **EWB** соберем схему простейшего сдвигового регистра. Соединим четыре D-триггера последовательно друг за другом так, что сигнал с выхода Q_1 первого будет подаваться на вход D_1 второго триггера

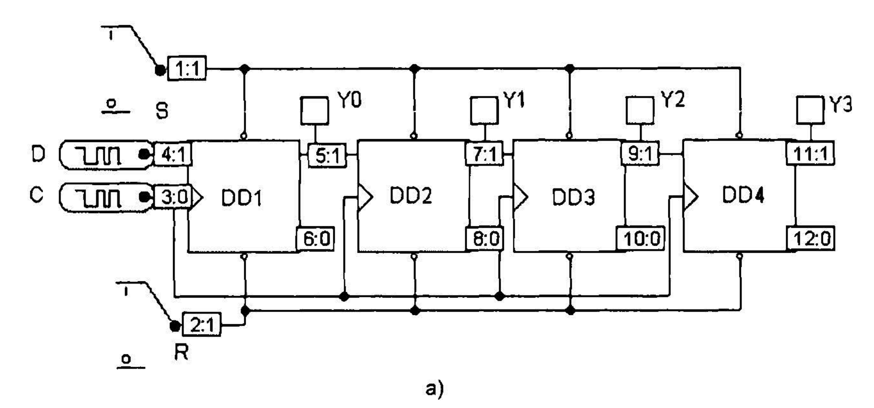
и т.д. (рис. 141,а); данные, т.е. 1 или 0 в соответствующем разряде, будем вводить клавишей [D]; на выводы синхронизации всех триггеров C1 одновременно будем подавать тактовые импульсы от клавиши [Space] Следя за состояниями поразрядных индикаторов Y0–Y3, будем судить о работе регистра Пусть требуется ввести в регистр число $5_{10} = 0101_2$ Начнем с нулевого состояния (Y0=Y1=Y2=Y3=0). Подадим на вход первого триггера D=1 и [Space] =1 и включим моделирование: получим Y0=1. Если дважды три раза нажать на [Space], то все разряды постепенно заполнятся единицами, но это будет $1111_2 = 15_{10}$, а не то, что требовалось. Очистим регистр. Это можно выполнить разными способами, из которых мы выберем, пожалуй, наиболее экзотический. Не выключая моделирование, сделаем D=0 и [Space] =0 и, нажимая многократно [Space], «продвинем» 0 по всем разрядам так же, как до этого «двигали» 1. По наблюдаемой картине и дали название этого устройства: сдвиговый регистр На каждом новом тактовом импульсе он принимает одну новую цифру, сдвигая ранее запомненные (в том числе и нули) на один разряд, чтобы поместить новую Движение в регистре цифр напоминает движение людей через турникет (такты) на эскалатор метро по одному (1) друг за другом с интервалами (0) Не случайно подобные схемы называют на английском FIFO (First-In-First-Out, т е. первым вошел – первым вышел). Очевидно, что полученные результаты обусловлены описанной ранее работой D-триггера, выход Q которого повторяет входной сигнал D, пока C=1, и запоминает логический уровень входного сигнала при C=0 («прозрачная защелка»). Отсюда понятно, что для решения поставленной задачи надо, манипулируя клавишами D и [Space], аналогично описанному, «продвигать» в регистр ту последовательность, которая задана: 0101. Если использовать в подобном регистре D-триггеры с предустановками по S и R, то пользуясь ими можно установить единицы во все разряды сразу или очистить весь регистр. Соответствующая схема показана на рис. 141,в Для удобства наблюдения далее вместо логических индикаторов включим семисегментный индикатор с дешифратором HL1 (рис 141,в) При согласованном включении входных разрядов индикатора и выходных разрядов регистра, занесение в регистр числа 01010_2 даст на дисплее цифру 5 (см рис. 141,в)

В программе **МС** повторим схему, аналогичную рис 141,б, с тем лишь отличием, что в качестве источников сигнала данных D и тактового C, используем генераторы одноразрядного цифрового сигнала Stim1 (см рис 142,а), задав их в следующем формате:

.define C +0NS 1 +50NS 0 +100NS 1 +150NS 0 +200NS 1 +250NS 0 +300NS 1 +350NS 0 +400NS 1

.define D +0NS 0 +50NS 1 +100NS 0 +150NS 1 +200NS 0 +250NS 1 +300NS 0 +350NS 1 +400NS 0

В результате проведенного анализа на осциллограммах (см. рис. 142,б), при заданных сигналах (d3, d4), видно заполнение разрядов регистра «волной» единиц (d5, d7, d9, d11).



а)

RG1.CIR

d(3)
d(4)
d(5)
d(7)
d(9)
d(11)

0.000n 100.000n 200.000n 300.000n 400.000n 500.000n
T

б)

Рис. 142 (начало)

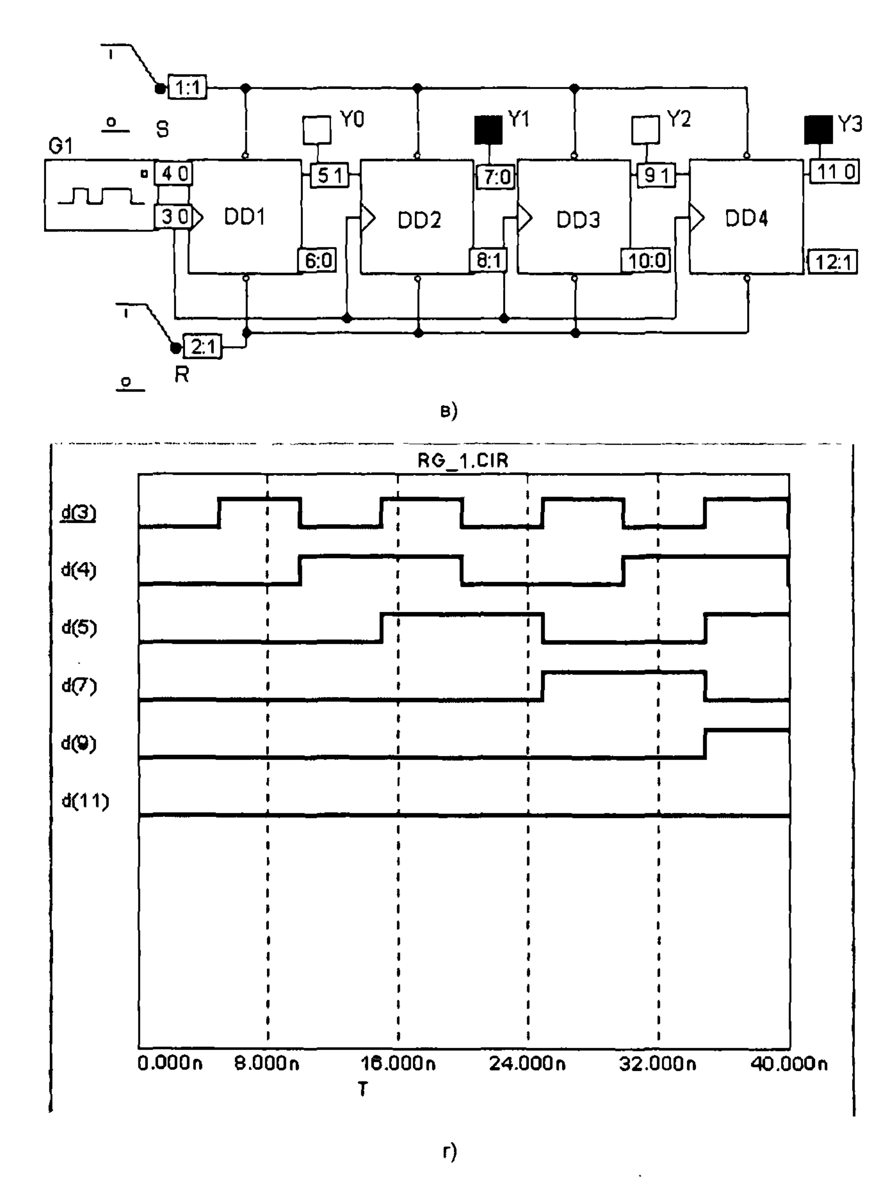


Рис. 142 (окончание). Четырехразрядный сдвиговый регистр (**MC**)

Для того чтобы занести в регистр определенное число, например 5_{10}, зададим входные сигналы от генератора двухразрядного цифрового сигнала G1 (см рис 142,в), который зададим в следующем формате:

```
DEFINE G1
+ label=begin
+ +0NS 00
+ +5NS 01
+ +5NS 10
+ +5NS 11
+ +5NS goto begin -1 times
```

и ограничим время наблюдения интервалом 40NS. Результат моделирования виден на рис. 142,в на индикаторах (0101) и графиках рис. 142,г.

В рассмотренном регистре данные вводятся последовательно через единственный вход, а выводятся в параллельном виде на выходах разрядов регистра Такая процедура широко используется, например, для преобразования битов программы, считанных с диска компьютера, в параллельный код для ввода в основную память.

При необходимости с регистра можно вывести информацию и в инверсной форме, если воспользоваться инверсными выходами триггеров.

Введя в схему регистра дополнительную комбинационную логику, можно создать реверсивный регистр с управляемым направлением сдвига Для этого в разрядные схемы вводятся элементы 2×2 *И-ИЛИ-НЕ*, которые выполняя роль мультиплексора, изменяют направление передачи сигнала. Соответствующий субблок показан в развернутом виде на рис. 143,а и в виде схемного компонента MUX на рис 143, б На основе этих компонентов и D-триггеров собран четырехразрядный реверсивный сдвигающий регистр (см рис 143, в)

При S=1 выполняется сдвиг информации вправо (от регистров младших разрядов к старшим), а при S=0 – влево (от старших – к младшим). Надо иметь в виду, что понятия «лево» и «право» относятся здесь не к общепринятой системе упорядоченной записи чисел, а к схеме где, как правило, слева вход, а справа – выход Поэтому, разряды регистра расположены инверсно по отношению к записи чисел. Итак, если задавать из указанного на рис 143,в состояния тактовые импульсы (клавишей Space), то единицы будут двигаться назад к входу, пока не заполнят все разряды. Если же в этом состоянии включить S=1, то нули будут двигаться от входа, пока не очистят весь регистр Для удобства отсчетов двоичных чисел можно инвертировать геометрическое расположение индикаторов, не изменяя существа схемы их соединения. При инверсном расположении индикаторов (см. рис. 143,г) наблюдаемая картина как бы входит в противоречие с понятиями «лево» и «право». Однако при маркировке микросхем принята именно такая нумерация, хотя в литературе, особенно отечественной, встречаются и иные варианты.

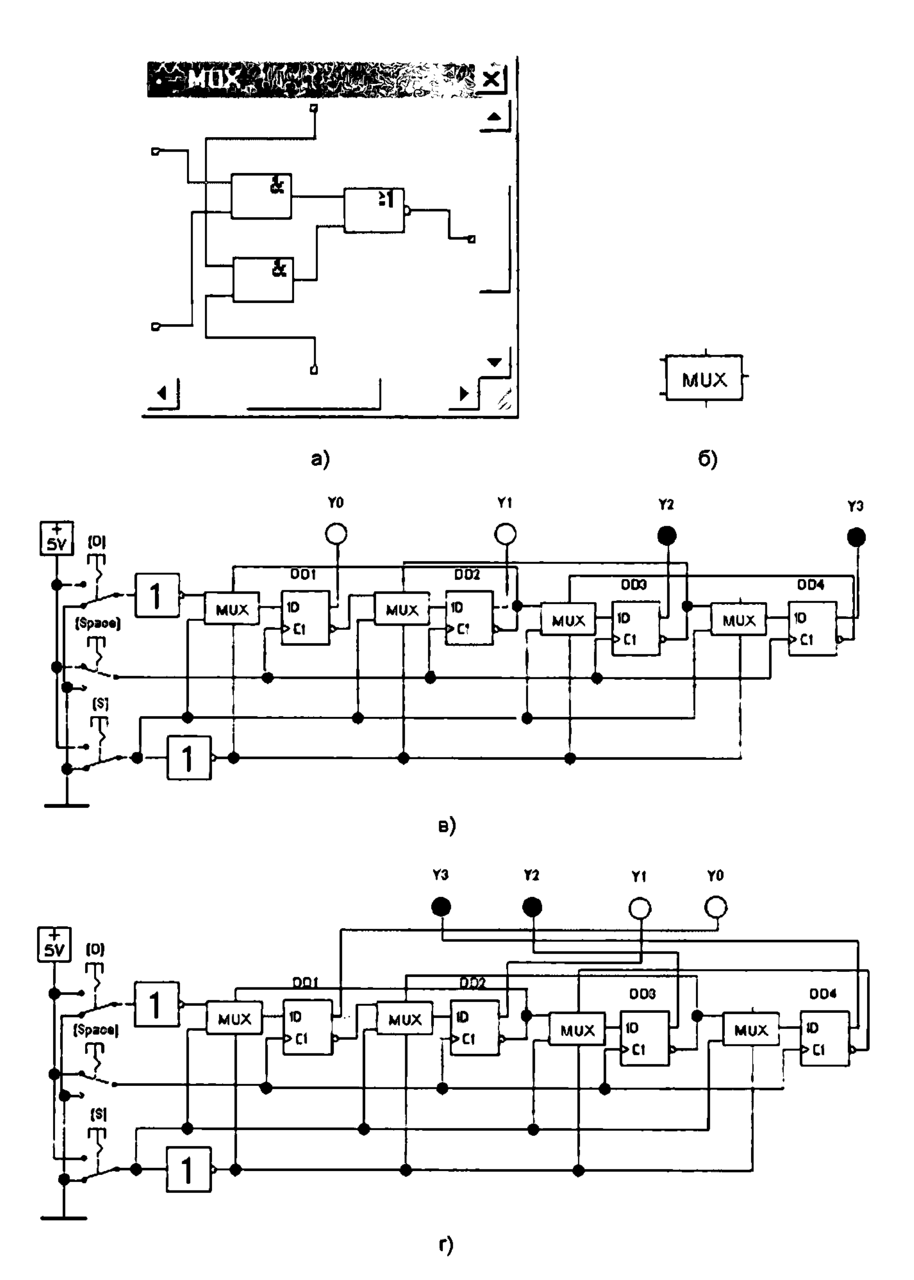


Рис. 143. Четырехразрядный реверсивный сдвигающий регистр (**EWB**)

Параллельные регистры хранения

В параллельных (статических) регистрах схемы разрядов не обмениваются данными между собой. Общими для разрядов обычно являются цепи тактирования, сброса/установки, разрешения выхода или приема, т е цепи управления В этих регистрах прием и выдача слов производятся по всем разрядам одновременно. В них хранятся слова, которые могут быть подвергнуты поразрядным логическим преобразованиям

Для того чтобы запомнить в регистре n-разрядное двоичное слово, потребуется n D-триггеров, поскольку каждый из них может запомнить один бит на фронте тактового сигнала. Если ограничиться четырехразрядным словом, то схема регистра в программе **EWB** примет такой вид (см рис. 144). Задав некоторое число ключами 1–4 и включив моделирование, после перевода клавиши Space в верхнее положение (переход от низкого уровня к высокому – фронт тактового импульса) получим это число в регистре. Если теперь еще раз нажать на клавишу Space, переведя ее в положение низкого уровня (спад тактового импульса), то ничего не изменится:

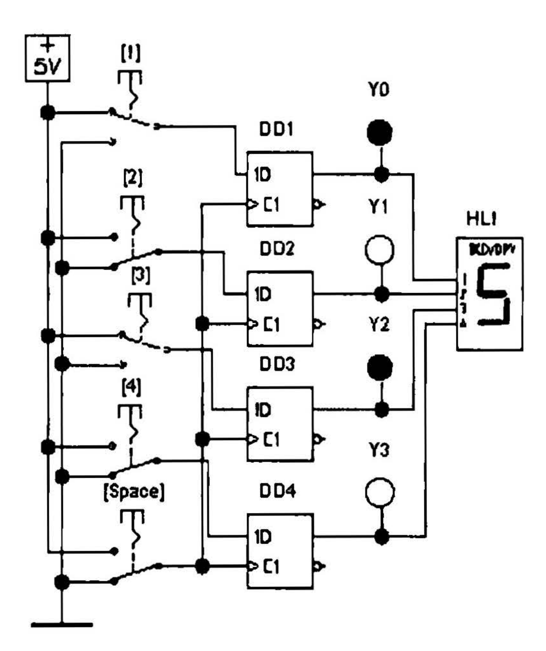
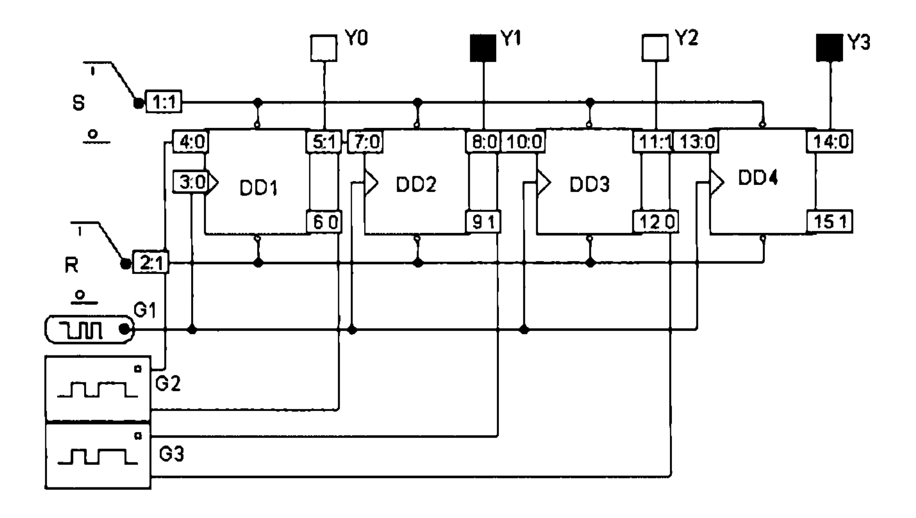


Рис. 144. Логическая структура четырехразрядного регистра хранения (**EWB**)



а)

RG3.CIR

d(3)
d(4)
d(5)
d(8)
d(11)
d(14)

0.000n 8.000n 16.000n 24.000n 32.000n 40.000n
T

б)

Рис. 145. Логическая структура четырехразрядного регистра хранения (**MC**)

занесенное число сохранится в регистре Оно останется там, если далее перевести клавиши 1 и 3 на низкий уровень. Занесенное число может храниться в этом регистре сколь угодно долго (при наличии питания). Поэтому подобные регистры относят к регистрам хранения. Изменение состояния регистра можно выполнить, изменив входные сигналы и подав новый тактовый импульс.

В программе **МС** соберем логическую структуру регистра хранения на D-триггерах с предустановками (см. рис. 145,а). Используем специальные генераторы цифровых сигналов (Stim) G1, G2 и G3, задав их в формате:

```
.DEFINE G1
+ +0NS 0
+ label=begin
+ +5NS 1
+ +5NS 0
+ +5NS goto begin -1 times

.DEFINE G2
+ label=begin
+ +0NS 00
+ +5NS 01
+ +5NS 10
+ +5NS 11
+ +5NS goto begin -1 times

.DEFINE G3
+ label=begin
+ +0NS 00
+ +5NS 01
+ +5NS 10
+ +5NS 11
+ +5NS goto begin -1 times
```

В результате моделирования получатся графики, показанные на рис 145,б

Совет (**МС**) Форматы заданий на моделирование генераторов цифровых сигналов типа Stim можно получить, скопировав подходящий генератор из файлов типовых примеров, прилагаемых к программе, и отредактировав их параметры. Для рассматриваемых схем их можно набрать

из книги в окно редактирования свойств генератора или в текстовую область создаваемой схемы. При этом надо четко идентифицировать позиционные обозначения и атрибуты команды описания сигнала.

Параллельные регистры сдвига

Последовательный регистр сдвига обладает двумя недостатками: он позволяет вводить только по одному биту информации на каждом тактовом импульсе и каждый раз при сдвиге информации вправо теряется крайний правый информационный бит Эти недостатки устраняются в параллельно загружаемых регистрах сдвига кольцевого типа. Кольцевое перемещение информации предусматривает возврат данных с выхода регистра обратно на его вход, что исключает потерю данных при сдвиге.

Построим логическую структуру четырехразрядного параллельного кольцевого регистра сдвига в программе **EWB**. Для этого используем JK-триггеры с предустановками (см. рис. 146). Особенностью этой схемы помимо параллельной загрузки данных ABCD является организация цепи обратной связи: выход Q триггера DD4 соединяется с входом 1J триггера DD1 и выход Q` триггера DD4 соединяется с входом 1K триггера DD1 (см рис 146)

При включении питания (режим начала моделирования) на выходах может установиться произвольная комбинация, в частности 1111. Очистка регистра (установка выходов в состояние 0000) осуществляется подачей лог. 0 на входы R. Поскольку загрузка данных производится через инверсные входы S, то при кратковременной подаче лог. 0 на входы в соответствующие разряды будет занесена лог. 1 (см. рис. 146). Подача тактовых импульсов (клавишей Space) приводит к кольцевому перемещению информации Нетрудно убедиться, что возврат в исходное состояние после загрузки будет в общем случае происходить за четыре такта. Разрыв контура обратной связи превращает кольцевой регистр в обычный сдвиговый регистр с параллельной загрузкой данных.

Помимо описанных выше регистров имеются также последовательно-параллельные регистры, имеющие входы–выходы одновременно последовательного и параллельного типа. Существуют также варианты регистров с возможностью любого сочетания способов приема и выдачи слов. Для современной схемотехники характерно построение регистров на D-триггерах преимущественно с динамическим управлением. Многие из них имеют выходы с третьим состоянием, некоторые регистры относятся к числу буферных, т. е. рассчитаны на работу с больши-

ми емкостными и/или низкоомными активными нагрузками. Это обеспечивает их работу непосредственно на магистраль (без дополнительных схем интерфейса).

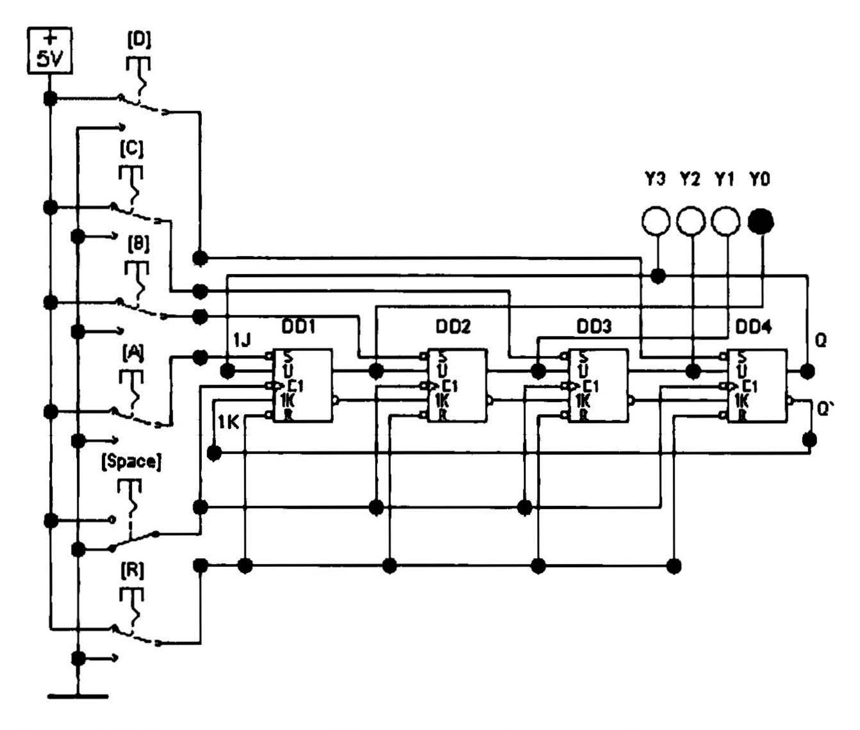
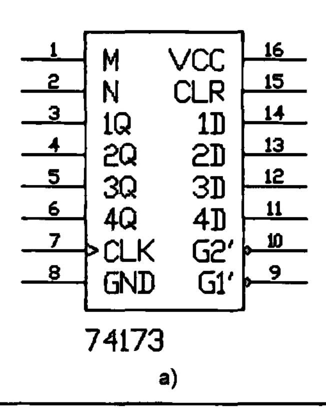


Рис. 146. Четырехразрядный параллельный кольцевой рвгистр сдвига (EWB)

Регистры на микросхемах

Начнем с рассмотрения четырехразрядного регистра на микросхеме ТТЛ 74173. В программе **EWB** войдем в меню выбора цифровых микросхем (Digital ICs) и далее в библиотеку моделей с номерами 741xx. Найдя нужную микросхему (см. рис. 147,а), выводим ее в центр рабочего экрана и щелкаем по ее УГО ПКМ, а затем – ЛКМ по позиции Help в появившемся подменю. На экране появится таблица истинности данной микросхемы (см. рис. 147,б). Как следует из ее ТИ и цоколевки (см. рис. 147,а), эта микросхема представляет собой четырехразрядный

статический регистр, построенный на D-триггерах с прямыми динамическими входами 1D...4D, входом сброса Clear и выходами 1Q...4Q с третьим (Z) состоянием, управляемым сигналом G1` и G2`. При обычной работе на управляющих выводах надо задать низкий уровень напряжения; аналогично низкий уровень напряжения подается на входы разрешения записи M и N. Вход сброса имеет высокий активный уровень. Стоящий в таблице символ POS – усеченное от POSITIVE – положительный, означает, что смена состояний D-триггеров происходит, когда напряжение тактового импульса нарастает от лог. 0 до лог. 1. Таким образом, данные поступают в регистр на фронте каждого тактового импульса CLC.



а)

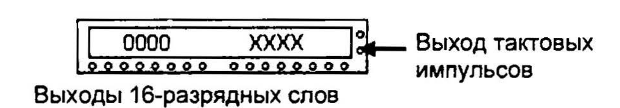
74173 (4-bit D-type Reg w/3-state Out)

D-type Register Truth table:

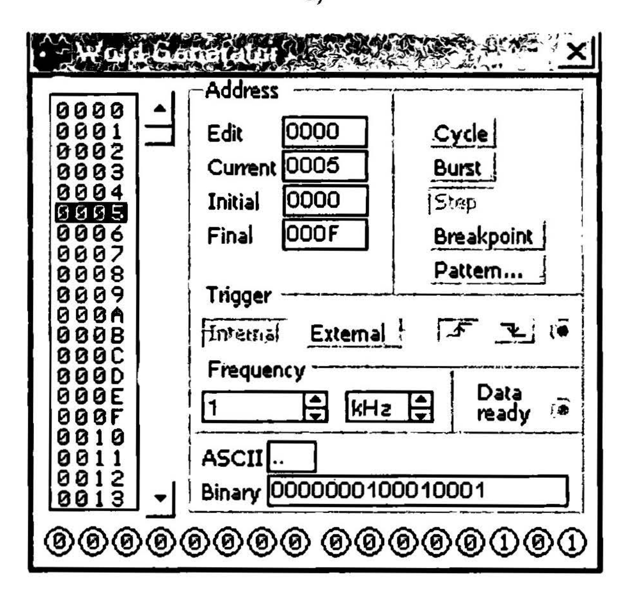
CLEAR	CLK	DATA ENABLE G'1	DATA ENABLE G'2	DATA D	OUTPUT Q
1	X	X	X	X	0
0	0	X	X	X	Q0
0	POS	1	X	X	Q0
0	POS	X	1	X	Q0
0	POS	0	0	0	0
0	POS	0	0	1	1

б)

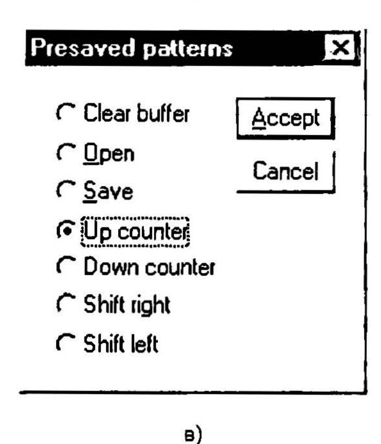
Рис. 147. Микросхема 74173 ((EWB)



а)



б)



в)

Рис 148 Генератор слов (**EWB**)

Для исследования регистра соберем схему, в которой в качестве источника данных будем использовать генератор слов (Word Generator).

Этот прибор выбирается в панели Instruments по его пиктограмме [01X 0000]. После двойного щелчка ЛКМ по схемному изображению генератора (рис. 148,а) откроется его лицевая панель с установочными и управляющими кнопками (см. рис. 148,б). На выходах генератора (рис. 148,а) можно получить коды шестнадцатиразрядных двоичных слов, выбираемых на пользовательской панели. Для набора слова надо щелкнуть ЛКМ в соответствующем разряде экранного буфера (заполненного нулями) и набрать с клавиатуры соответствующую цифру 0 или 1, стоящую в данном разряде. Дальше, как при печати таблиц, лучше пользоваться клавиатурой Все комбинации задаются в шестнадцатеричном коде. Номер редактируемой ячейки показывается в окошке Edit блока Address, при этом верхняя ячейка считается нулевой. Следующее окошко Current показывает номер текущей ячейки, кодовая комбинация с которой в данный момент поступает на выход генератора В окошках Initial и Final указываются соответственно номер начальной и конечной ячеек, в которые заносится информация. Занесение данных в выделенную ячейку можно выполнить в коде ASCII или бинарном коде (окошко Binary). При печати слов в этих кодах программа автоматически преобразует их в гексагональный код и заносит в редактируемую ячейку. Этим можно воспользоваться для занесения цифр шестнадцатеричного кода от A до F. При наборе слов в ячейках в шестнадцатеричном коде, можно увидеть их бинарное представление. 16 нижних окошек на лицевой панели служат для индикации состояний разрядов на выходе генератора в процессе его работы. В схемном компоненте генератора (рис. 148,а) этим показаниям соответствуют уровни напряжения на 16 клеммах. С этих клемм поразрядные сигналы по подключенным к ним проводам подаются на соответствующие цифровые узлы. В правой части этого схемного компонента мы будем использовать также нижнюю из двух клемм для получения тактовых импульсов. Частота следования тактов задается на лицевой панели в окошке Frequency (с учетом единиц измерения Hz, kHz, MHz). Генератор может работать с этой частотой при нажатии на кнопку Cycle или с заранее выбранного слова до конца при нажатии на кнопку Burst. Очень удобным является также пошаговый режим (Step), заменяющий цифровые ключи. Кнопка [Breakpoint] (дословно – точка разрыва) позволяет останавливать

работу генератора на заранее выбранной ячейке. Наконец, кнопка Pattern... (картина) при нажатии на нее ЛКМ вводит ряд дополнительных возможностей, показанных на открывающейся панели Presaved pattern (предварительно сохраненная картина). Здесь мы пока воспользуемся возможностью Up counter (прямого счета), автоматически заносящей в экранный буфер кодовые комбинации, начиная с 0000 до FFFF (65535_{10}).

Совет При проведении моделирования с цифровыми устройствами полезно иметь перед глазами таблицу соотношений между десятичными, двоичными и шестнадцатеричными числами в пределах $0–15_{10}$, например, для этого можно воспользоваться левой половиной таблицы 3. В случае использования больших чисел целесообразно выполнять переводы из одних кодов в другие на компьютере, обратившись к калькулятору, входящему в пакет стандартных программ Microsoft Office.

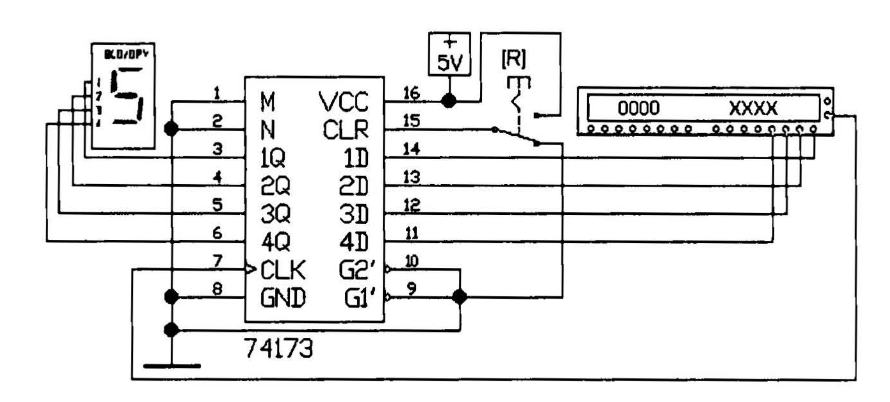


Рис. 149. 4-разрядный статический регистр на микросхеме 74173 (**EWB**)

Собрав схему по рис. 149, проведем испытания регистра. Открыв лицевую панель генератора слов, сделаем на ней необходимые установки и выберем пошаговый режим (см. рис. 148,б). Включаем моделирование и, нажимая многократно ЛКМ клавишу Step, шаг за шагом загружаем в регистр выбранные слова, что соответствует двум нижним строкам ТИ (рис. 147,б). Очистку регистра можно произвести ключом [R], кратковременно подав на вход CLC лог. 1 (см. верхнюю строку ТИ на рис. 147,б). Если

в схему добавить управление по входам разрешения записи M, N и выхода G1`, G2`, введя дополнительные ключи или специальные кодовые сигналы от генератора, то можно исследовать и другие режимы работы регистра

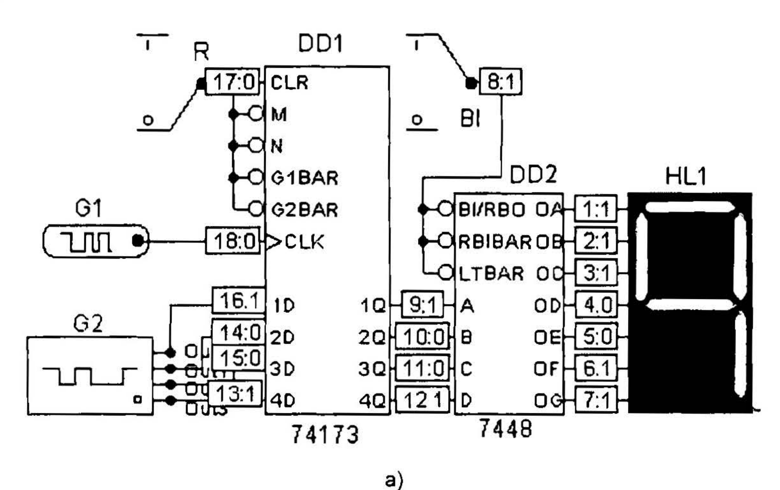
Исследуем работу этой же интегральной схемы в программе **MC**. Выберем из цифровой библиотеки компонентов микросхему 74173 и будем использовать следующие генераторы цифровых сигналов: типа Stim1 – генератор одноразрядного цифрового сигнала как тактовый и Stim4 – генератор четырехразрядного цифрового сигнала в качестве генератора данных, задав их в следующем формате:

```
.DEFINE G1
+ +0NS     0
+ label=begin
+ +250NS 1
+ +250NS 0
+ +250NS goto begin -1 times

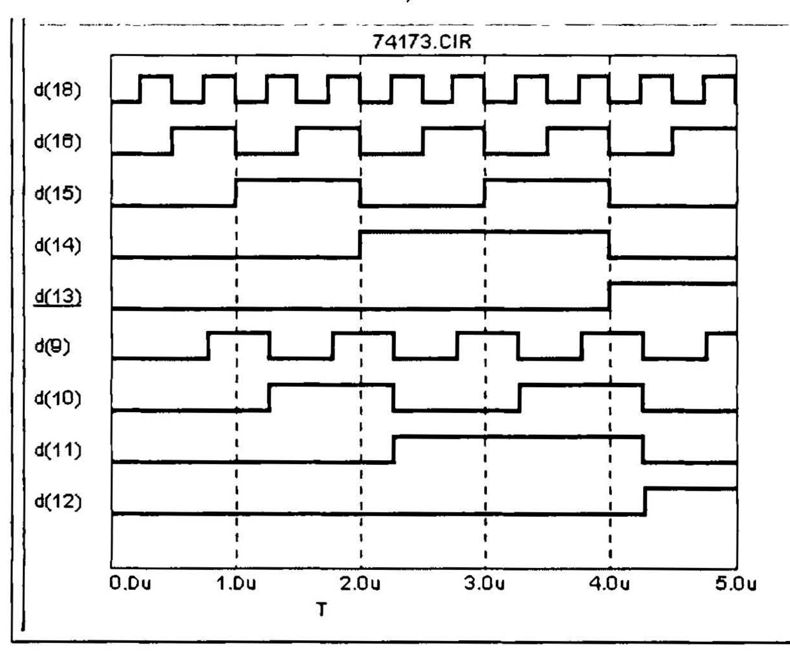
.DEFINE G2
 +0NS 0
 +label=start
+500NS incr by 1
+1000NS goto start until GE F
```

Для индикации выходов регистра по-прежнему будем использовать дешифратор двоичного кода 7448 в сборке с семисегментным индикатором (см. рис. 101). Собранная схема показана на рис. 150,а и, соответственно, на рис. 150,б – графики состояний ее цифровых узлов. Здесь надо отметить, что как состояния цифровых узлов, так и показание индикаторного дисплея соответствуют состоянию регистра в конечный момент времени моделирования. На рис. 150 – это 5.0u = 5 микросекунд. (Напомним, что в программе **MC** дольная приставка «микро» обозначается латинской буквой «u», а основная единица измерения (s – секунда) опускается; десятичным разделителем служит не запятая, а точка.) Графики двоичных сигналов на рис. 150,б сверху вниз таковы: d(18) – тактовые импульсы; d(16)...d(13) – входы регистра; d(9)...d(12) – выходы регистра Если сделать отсчет по графику для входных и выходных сигналов в момент 5.0u (по его ординате), то с учетом расположения разрядов в обоих случаях получим 1001(9 – на индикаторе после декодирования) Если же в меню анализа Transient выбрать другое время окончания моде-

лирования, то, как это следует из приведенных графиков, отсчет будет иным, например для 3.0u получим $0101_2=5_{10}$.



а)



б)

Рис. 150. Четырехразрядный статический регистр на микросхеме 74173 (**MC**)

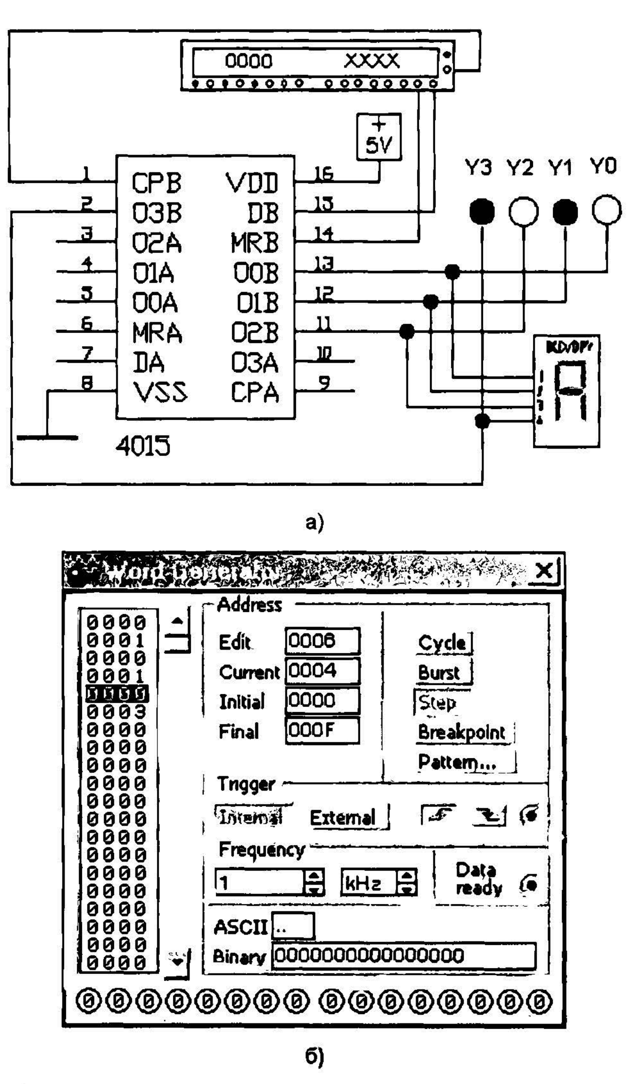
Примером четырехразрядного статического сдвигового регистра на микросхеме КМОП может служить микросхема 4015. В программе **EWB** имеется подборка микросхем сдвиговых регистров, находящаяся в библиотеке Digital по пиктограмме SRG (SRG – сокращение от Shift REGister – сдвиговый регистр) Открывающаяся библиотека регистров показана на рис. 151 Выберем необходимую микросхему. Она имеет в одном корпусе два четырехразрядных регистра, отмаркированных соответственно буквами А и В. Все триггеры в данной микросхеме двухступенчатые D-типа. Соберем схему на регистре В (см. рис. 152,а). Для удобства интерпретации результатов выход регистра будем контролировать и в цифровой форме и поразрядно логическими пробниками. В качестве данных будем использовать выход самого младшего разряда генератора слов, подав сигнал с соответствующей клеммы на последовательный вход DB (Data B) Тактовые импульсы с генератора подадим на вход синхронизации CPB (Clock Pulse B). Регистр имеет вход асинхронного сброса MRB (Mode control Reset B0), действующего при лог. 1. Мы соединили его с выходом второго разряда генератора, поэтому, задавая в определенной ячейке двоичный выход в этом разряде лог 1, будем автоматически получать сброс регистра. Наберем следующие кодовые последовательности: 0000, 0001, 0000, 0001, 0000, 0003. Нажмем клавишу Step. Включим моделирование на дисплее загорится 0, индикаторы не горят. Нажмем Step дисплей – 1, индикаторы – 0001. Еще раз Step: дисплей – 2, индикаторы – 0010 Далее, Step: дисплей – 5, индикаторы – 0101. Продолжаем, Step дисплей – А, индикаторы – 1010. Последний раз –Step дисплей – 0, все разряды обнулены и индикаторы – 0000

Shift Registers

- Generic 4-bit Univ Shift Reg
- 4014 (8-bit Static Shift Reg)
- 4015 (Dual 4-bit Static Shift Reg)
- 7491 (8-bit Shift Reg)
- 74164 (8-bit Parallel-Out Serial Shift Reg)
- 74165 (Parallel-load 8-bit Shift Reg)

Accept | Cancel

Рис. 151. Окно выбора библиотечных моделей регистроа (**EWB**)



а)

б)

Рис. 152. Четырехразрядный сдвиговый регистр на микросхеме 4015 (**EWB**)

Среди различных регистров в интегральном исполнении выделяется однокорпусная микросхема 74194, представляющая собой четырехразрядный двунаправленный универсальный регистр сдвига. Этот регистр позволяет сдвигать информацию

и влево, и вправо, загружать данные как последовательно, так и параллельно. Регистр можно каскадировать и использовать для кольцевого перемещения информации.

74194 (4-bit Bidirectional Univ. Shift Reg)

CLEAR	MODE S1	MODE S0	CLK	SERIAL LEFT	SERIAL RIGHT	PARALLEL A	PARALLEL B	PARALLEL C	PARALLEL D	OUTPUTS QA	OUTPUTS QB	OUTPUTS QC	OUTPUTS QD
0	X	X	X	X	X	X	X	X	X	0	0	0	0
1	X	X	0	X	X	X	X	X	X	QA0	QB0	QC0	QD0
1	1	1	POS	X	X	a	b	c	d	a	b	c	d
1	0	1	POS	X	1	X	X	X	X	1	QAn	QBn	QCn
1	0	1	POS	X	0	X	X	X	X	0	QAn	QBn	QCn
1	1	0	POS	1	X	X	X	X	X	QBn	QCn	QDn	1
1	1	0	POS	0	X	X	X	X	X	QBn	QCn	QDn	0
1	0	0	X	X	X	X	X	X	X	QA0	QB0	QC0	QD0

а)

б)

Рис. 153. Универсальный регистр сдвига на микросхеме 74194 (**EWB**)

В программе **EWB**, войдя в библиотеку регистров, выбираем микросхему 74194 и, обратившись к предметной помощи, открываем ТИ этого регистра (см. рис. 153,а). Анализ этой таблицы показывает, что если на вход (CLEAR)' поступает напряжение низкого уровня, то на выходах QA...QD также устанавливается низкий уровень независимо от состояний всех других входов (они помечены меткой безразличного состояния x). При подаче на вход (CLEAR)' напряжения низкого уровня, режим работы регистра определяется состояниями входов S1 и S0. Данные, поступающие последовательно на вход SL (Serial Left – последовательно влево), когда на вход S0 подается напряжение низкого уровня, а на вход S1 – напряжение высокого уровня (S1=1,S0=0), сдвигаются влево (в соответствии с принятым выше определением понятий «лево» и «право») на каждом фронте тактового импульса CLC (на переходе – POS). Сдвиг данных вправо происходит, когда на вход S0 подается напряжение высокого уровня, а на вход S1 – напряжение низкого уровня (S1=0, S0=1), при этом данные поступают последовательно на вход SR (Serial Right – последовательно вправо). Синхронная параллельная загрузка данных через входы A D производится при S1=S0=1. В это время последовательные вводы и сдвиги запрещены (соответствующие порты заперты). Заканчивая рассмотрение таблицы, укажем, что дополнительные индексы 0 и n, которыми помечены выходы Q, означают их состояния соответственно до и после прохождения тактовых импульсов.

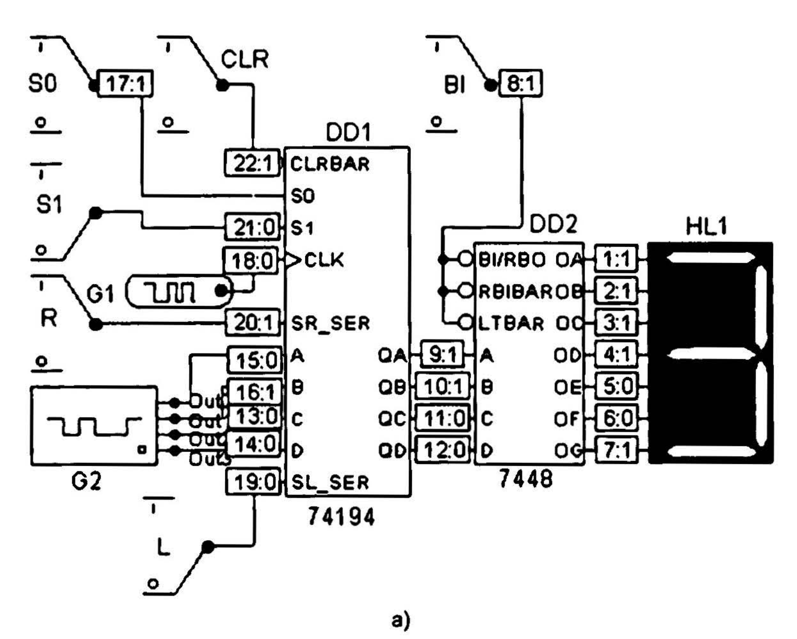
В сжатом виде таблица операций такова:

S1	S0	Операция
0	0	Хранение
0	1	Сдвиг вправо
1	0	Сдвиг влево
1	1	Параллельная загрузка

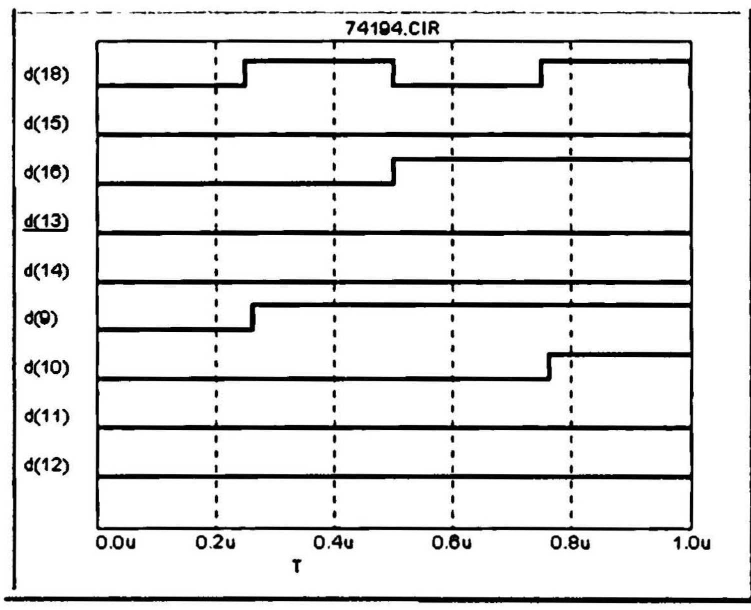
Собранная схема (см. рис 153,б) не имеет по существу никаких особенностей, кроме того, что данными, которые загружаются через последовательные входы, являются сами тактовые импульсы. Это сделано для упрощения схемы и анализа ее работы и может быть изменено произвольным образом. Перед началом моделирования откроем лицевую панель генератора слов и, воспользовавшись опцией Up counter, занесем в экранный буфер

кодовые комбинации прямого счета (см рис. 148,б), ограничим пределы кодовых комбинаций от 0000 до 000F и выберем пошаговый режим работы. Теперь, задавая ключами [0] и [1] различные состояния управляющих входов S0 и S1, проверяем по таблице истинности все случаи загрузки и сдвига Практика показывает, что это занятие (с учетом использования двоичной и шестнадцатеричной систем и выбранных определений понятий «лево» и «право») не менее занимательно, чем разгадывание головоломок, кроссвордов и шахматных этюдов. Особенно интересно предугадать новое состояние регистра при смене режима его работы без очистки разрядов. Имейте в виду, что выключение моделирования приводит к очистке регистра, а его включение в данной схеме, при подключенных последовательных входах, сразу дает единицу в соответствующем разряде слева или справа. На приведенном рисунке (рис. 153,б) S0=0 и S1=1, т.е. включен режим сдвига влево До включения моделирования регистр пуст: индикаторы – 0000, дисплей – 0. При включении моделирования показания будут таковы: индикаторы – 1000, дисплей – 8; Step1 индикаторы – 1100, дисплей – C, Step2 индикаторы – 1110, дисплей – E, Step3. индикаторы – 1111, дисплей – F. Регистр заполнился за четыре такта (считая включение) и дальнейшие шаги не изменяют его состояния, так как на смену уходящей единице приходит новая. Если после шага Step1: индикаторы – 1100, дисплей – C (см. рис. 153,б), не выключая моделирование, поменять положение ключей и задать S0=1 и S1=0, то после очередного шага единица самого старшего разряда исчезнет, но ее место тут же займет единица из предыдущего разряда, одновременно в самом младшем разряде появится единица. В результате получим: индикаторы – 1001, дисплей – 9. Если после этого, не выключая моделирование, поменять положение ключа 1, сделав S0=1 и S1=1, а далее осуществить загрузку, то результат будет таким: индикаторы – 0011, дисплей – 3. Поскольку мы находились в ячейке 0002 генератора слов, а перешли на данном шаге и загрузили следующую ячейку 0003.

Соберем аналогичную схему в программе **MC**, воспользовавшись предыдущими заготовками из файла, схема из которого показана на рис. 150,а, заменив в нем микросхему DD1 и дополнив ключами (см. рис. 154,б). При заданном на рисунке положении ключей (S0=1, S1=0, R=1, L=0), происходит сдвиг информации от младших разрядов к старшим и за отмеченное время «набегает» два младших разряда: $0011_2=3_{10}$.

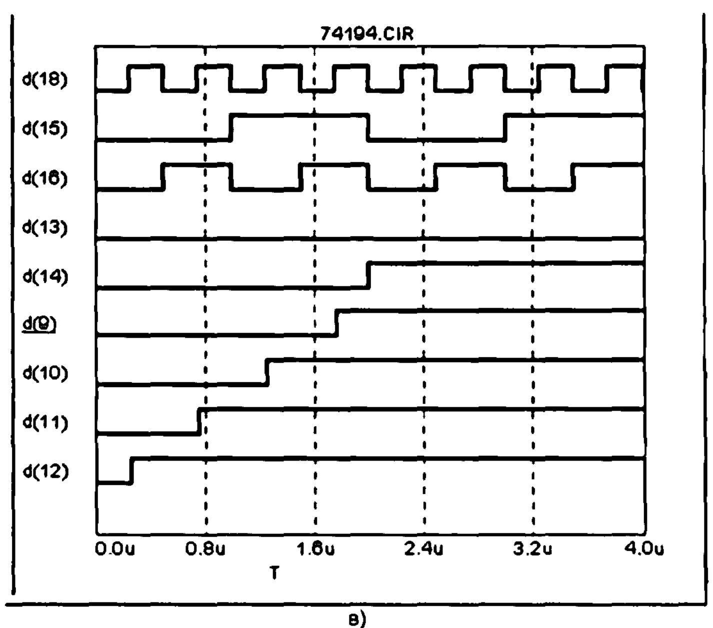


а)

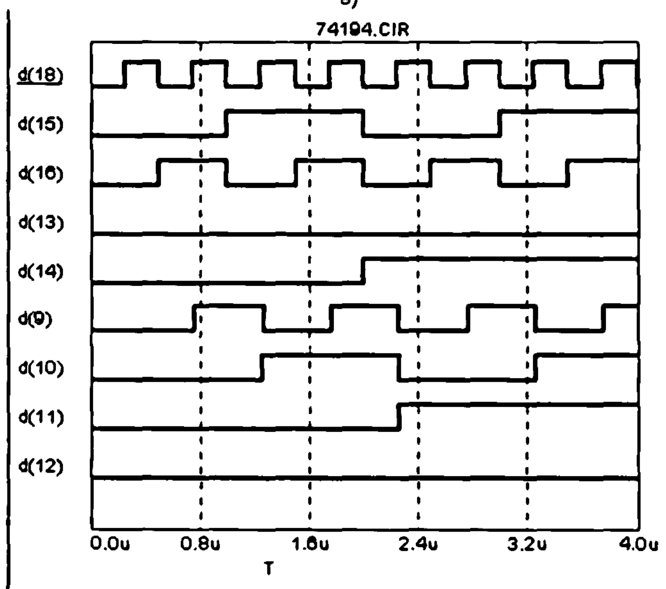


б)

Рис. 154 (начало)



в)



г)

Рис. 154 (окончание). Универсальный регистр сдвига на микросхеме 74194 (MC)

Поменяв положение ключей (S0=0, S1=1, R=0, L=1), получим движение информации в обратном направлении. На рис.154,в хорошо видно, как на нижних графиках (выходные сигналы) «волна» единиц перемещается от старших разрядов к младшим. Дисплей в конце моделирования будет темным, что соответствует в данной программе кодировке цифры F. Наконец можно выставить режим параллельной загрузки (S0=1 и S1=1) и увидеть, что за время 4 0u загрузится число $0111_2=7_{10}$ (см графики на рис. 154,г)

3.3. Счетчики

Бог создал натуральный ряд,
остальное – дело рук человеческих
Л. Кронекер (нем. математик XIX в.)

Счетчик – это устройство, которое подсчитывает число импульсов, поступивших на его вход за определенное время. Основной статической характеристикой счетчика является модуль счета M (емкость), который определяет число его возможных состояний. После поступления на счетчик M входных сигналов обычно начинается новый цикл, повторяющий предыдущий Основной динамической характеристикой счетчика является его быстродействие.

Счетчики классифицируют: по способу кодировки внутренних состояний (двоичные, счетчики Джонсона и др.); по направлению счета (суммирующие – прямого счета (Up-counter), вычитающие – обратного счета (Down-counter) и реверсивные (Up/Down counter), по принципу действия (синхронные и асинхронные); по способу организации межразрядных связей (с последовательным, параллельным и комбинированным переносом).

Асинхронные счетчики

Возможность подсчета числа импульсов уже как бы «генетически» заложена в устройство триггера. В принципе сам Т-триггер уже является счетчиком по модулю 2, так как он имеет два различных состояния и считает до двух: 0,1. Поэтому простейший двоичный суммирующий счетчик можно реализовать последовательной цепочкой счетных Т-триггеров (выполненных на D-триггерах, как было показано на рис.135,а). Для этого соединим их так, чтобы счетный вход очередного триггера подключался к инверсному выходу предыдущего (см рис 155,а)

В качестве задающего генератора импульсов применим ранее использованный функциональный генератор, сохранив прежние установки (см. рис. 136,б). Включив моделирование, наблюдаем за состоянием индикаторов счета. Инвертировав их расположение, делаем отсчет импульсов в двоичном коде (рис. 155,б), а, дополнив схему дисплеем, – в шестнадцатеричном коде (в котором цифры совпадают с десятичными до 9). Наконец, временные особенности счета можно увидеть на графиках, регистрируемых логическим анализатором (см. рис. 155,в,г) Установки анали-

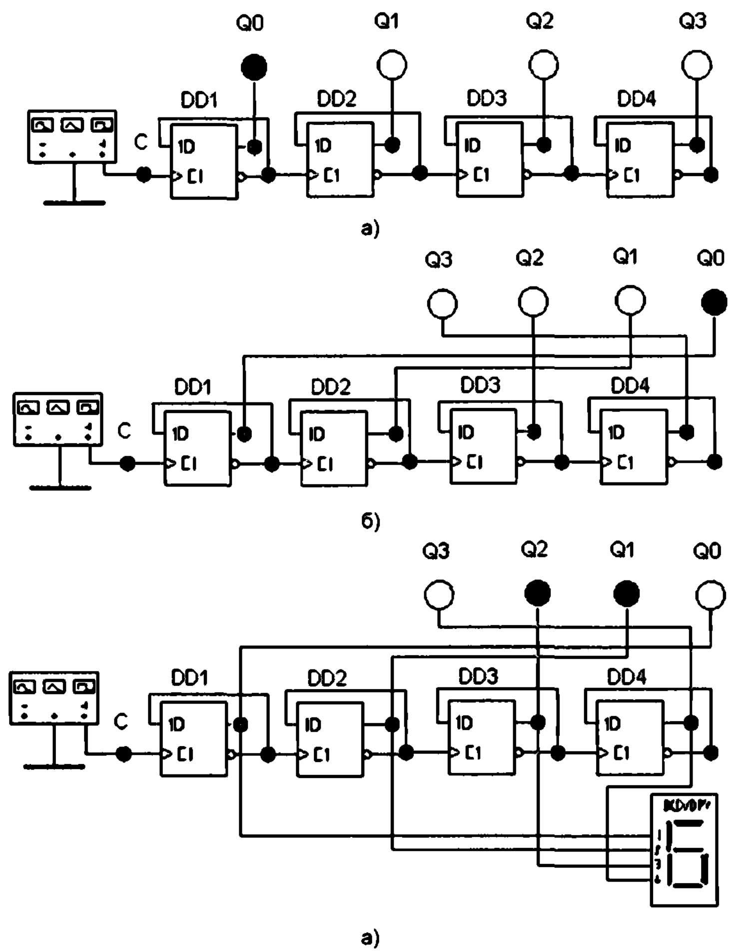
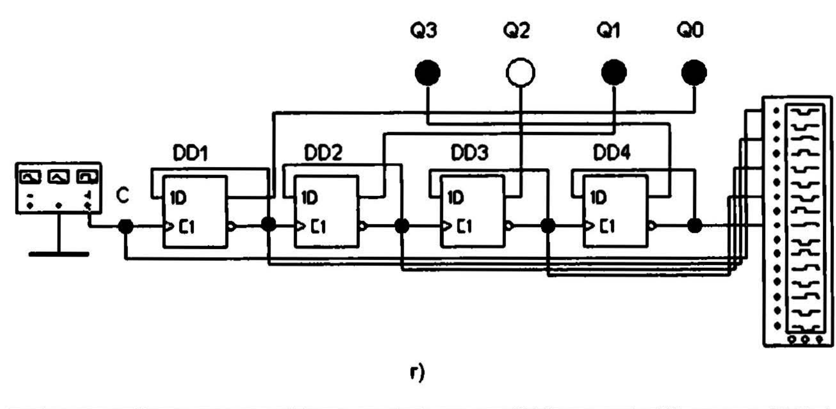


Рис 155 (начало)



г)

Logic Analyzer

Stop
Reset
T1 437.5000 ms 0000
T2 16.4375 s 0000
T2-T1 16.0000 s
Clocks per division 2
Clock
Set...
External
Qualifier
Trigger
Set...
Qualifier

д)

Рис. 155 (окончание). Суммирующий двоичный счетчик (**EWB**)

затора выполняем согласно рис. 138,г, увеличив только время развертки (Clocks per division) до 2 На графиках (рис. 155,г) видно, как первый триггер делит частоту входного сигнала на два, следующий делит также на два эту новую частоту и т.д. В результате, чтобы образовалась единица в последнем разряде этого счетчика на выходе первого триггера пройдет 8 импульсов, второго – 4, третьего – 2, четвертого – 1. Данный счетчик имеет 16 возможных состояний: от 0000 до 1111. Значит это счетчик по модулю 16 Он четырехразрядный по числу двоичных разрядов и асинхронный по запуску триггеров в этих разрядах. Поскольку импульсы передаются последовательно от одного триггера к другому, то подобные счетчики называют также счетчиками со сквозным переносом, или асинхронными, так как с приходом каждого импульса с задержкой относительно друг друга переключаются сразу несколько триггеров.

В рассматриваемой простой структуре возможна другая схема построения суммирующего счетчика: из предыдущего триггера переносный сигнал к счетному входу последующего триггера берется не с инверсного выхода, а с прямого. В этом случае выводы к индикации состояний также надо поменять и брать уже не с прямых выходов, а с инверсных (см. рис. 156).

Два последних возможных варианта межразрядных связей двух соседних триггеров и выводов индикации в этой структуре приводят к вычитающим счетчикам (см. рис. 157,а,б).

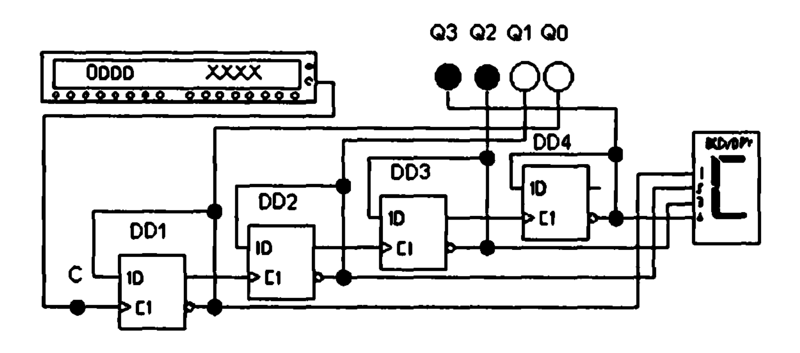


Рис. 156. Другой вариант суммирующего счетчика (EWB)

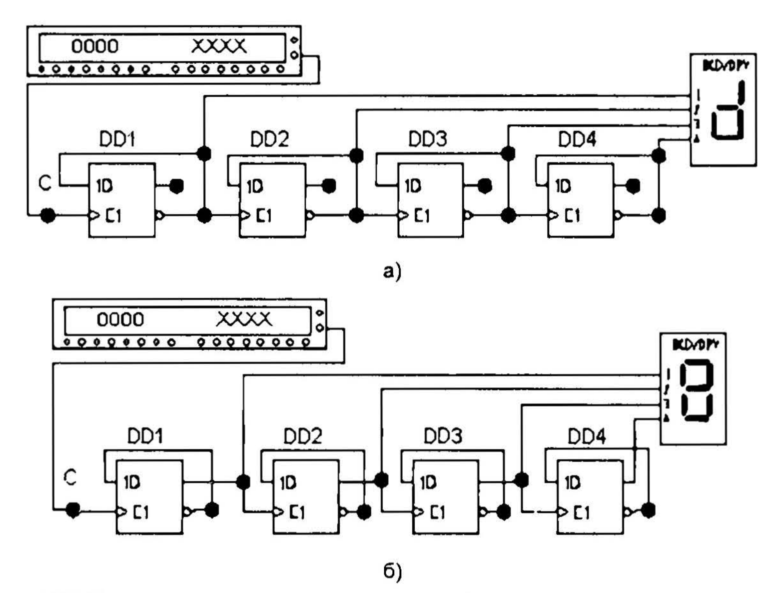


Рис. 157. Вычитающие двоичные счетчики (**EWB**)

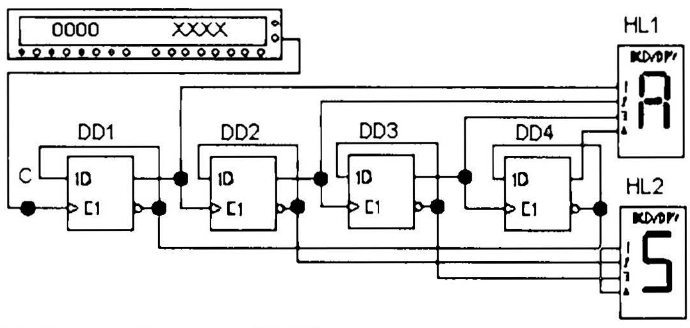
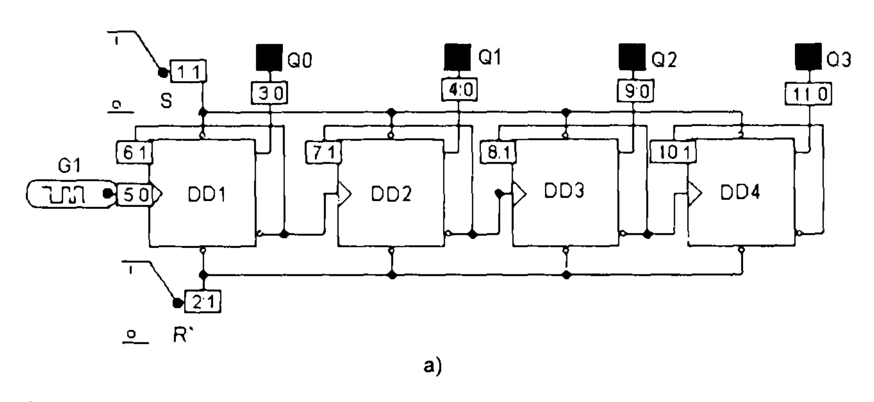


Рис. 158. Двоичный счетчик (**EWB**)

Можно также вывести сразу два дисплея от одного счетчика, но с противоположных выводов. Тогда на одном будет индицироваться прямой счет, а на другом обратный Сумма же показаний для рассматриваемого четырехразрядного счетчика по модулю 16, всегда равна 15. Для примера взят вычитающий счетчик с рис. 157 и к нему добавлен второй дисплей HL2 (см рис 158), на котором цифры нарастают по мере ввода импульсов, тогда как на HL1 они убывают Таким образом, для данной простой струк-

туры индикация выполненной переносной связи дает вычитающий счетчик, а противоположной – суммирующий.



а)

DCOUNTER1.CIR

d(5)
d(3)
d(4)
d(9)
d(11)

X0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
X1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
X2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
X3	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
№	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

0.000 3.200 6.400 9.600 12.800 16.000

T

б)

Рис. 159. Суммирующий двоичный счетчик (МС)

Схему суммирующего счетчика, аналогичную рис. 155,а, соберем в программе **MC** (см. рис. 159,а). Используем триггеры из файла, показанного на рис. 135,а. Генератор импульсов G1, выберем такой же, как на рис. 154,а, изменив формат его задания так, чтобы можно было сравнивать результаты обеих программ·

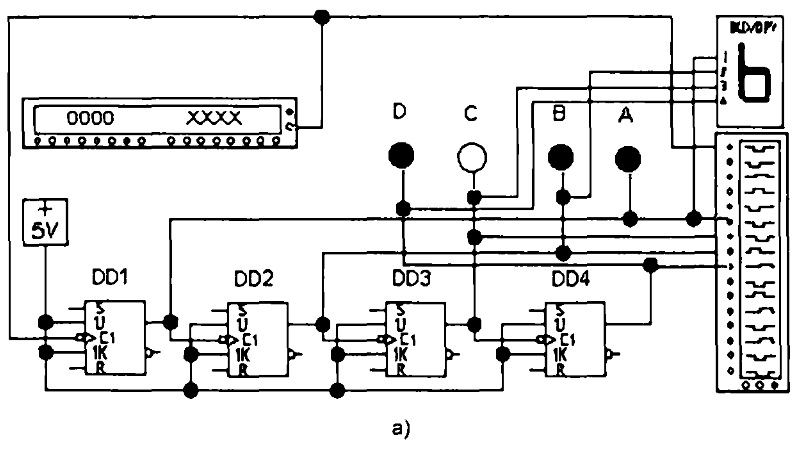
```
.DEFINE G1
+ +0S 0
+ label=begin
+ +0.5S
+ +0.5S 0
+ +0.5S goto begin -1 times
```

Таким образом, мы задали последовательность тактовых импульсов длительностью 0 5 секунды и периодом следования 1секунда, т.е. частотой следования 1Гц Здесь уместно отметить, что длительность такта – это период следования импульсов, а не длительность самого импульса (время, в течение которого значение напряжения на выбранном уровне больше единицы) Как правило, используют значительно более короткие по длительности импульсы, имеющие более крутые фронты. Корректно также говорить именно о частоте следования импульсов, а не о «частоте импульсов». Последнее лишено смысла, так как одиночный импульс содержит сплошной спектр гармоник, а периодическая последовательность – не одну, а целый ряд гармоник (на жаргоне «частот»).

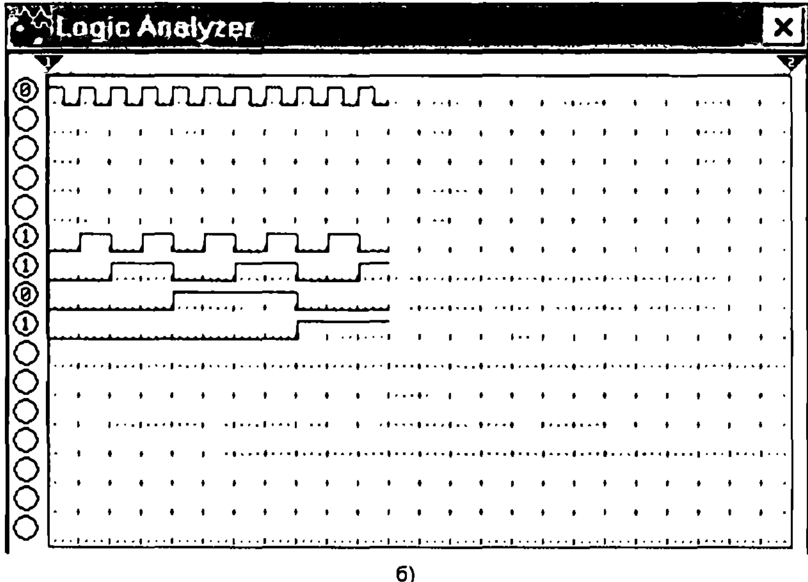
Результат моделирования в режиме Transient за время 16 с представлен на графиках (см. рис. 159,б) для следующих цифровых узлов: d(5) – вход; d(3) – выход Q0, d(4) – выход Q1, d(5) – выход Q2, d(11) – выход Q3. Сравнивая графики, полученные на логическом анализаторе (рис. 155,г) в программе **EWB** и при анализе Transient в программе **MC** (рис. 159,б), видим, что они однотипно отражают процесс последовательного деления частоты пополам при прохождении каждого разряда счетчика. Интересно сопоставить этот результат с таблицей шестнадцатеричных чисел, вмонтированной в нижней части графиков на рис. 159,б. Видно, что одному такту (из 8 нулей и 8 единиц) самого старшего разряда X3, соответствует два такта разряда X2 (из 4 нулей и 4 единиц), четыре такта (по 2 нуля и 2 единицы) разряда X1 и восемь тактов по одному нулю и одной единице в самом младшем разряде X0. (Собственно эта особенность и используется при заполнении таблицы на память.)

Схемы как суммирующего, так и вычитающего счетчиков можно построить на JK-триггерах, задав на информационных входах режим переключения (J=K=1) и выполнив соответствую-

щие переносные связи (см рис 160,а,161,а и 162,а). Сопоставляя графики изменения состояний разрядов на рис. 160,б, 161,б и 162,б видим, что процесс естественного счета вверх начинается с числа 0 в суммирующем счетчике и с числа $F_{16}=1111_2=15_{10}$ – в вычитающем. В последнем случае предусмотрена специальная предустановка всех разрядов в единицу ключом [S].



а)



б)

Рис. 160 Суммирующий счетчик на JK-триггерах (**EWB**)

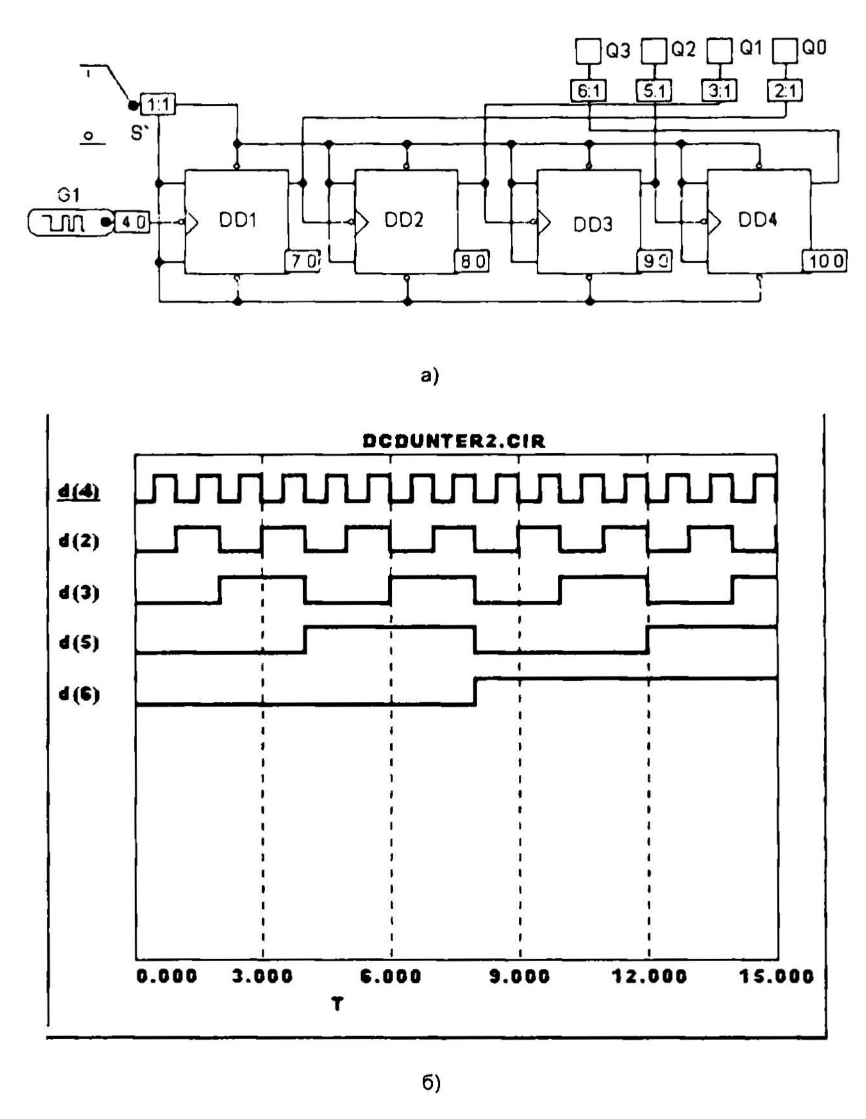
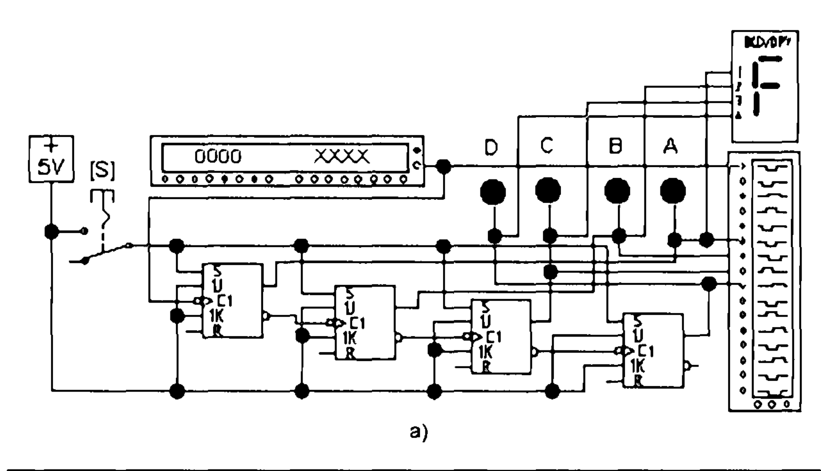
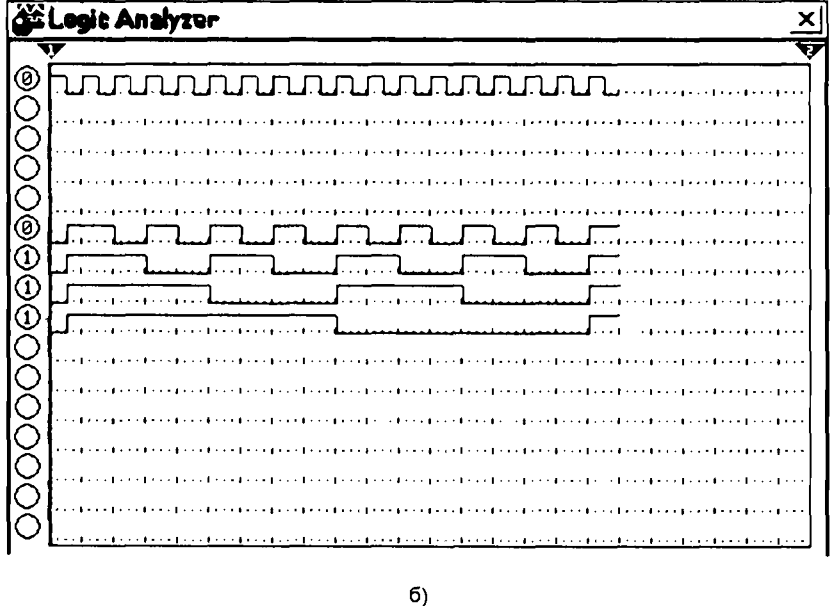


Рис. 161. Суммирующий счетчик на JK-триггерах (**MC**)



а)



б)

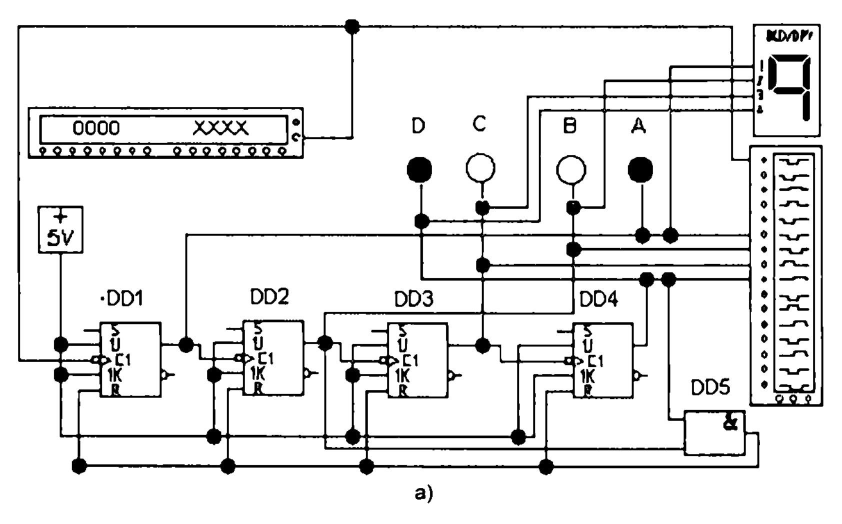
Рис. 162. Вычитающий счетчик на JK-триггерах (**EWB**)

Совет (**EWB**) Если при проведении моделирования окажется, что результаты счета, выводимые на показывающие устройства, не соответствуют необходимым по его конструкции, то, прежде всего, надо проверить правильность монтажа линий вывода и переноса. Полезно также сопоставить показания разных индикаторов: логических индикаторов, дисплея и логического анализатора. Не забывайте, что графический редактор программы может скрыть отсутствие реального контакта компонентов. При наложении друг на друга двух проводников они могут казаться идущими совсем по другому пути Поэтому желательно перейти к большим масштабам изображения и менее плотному монтажу, а также использовать цветную индикацию соединительных узлов и проводников

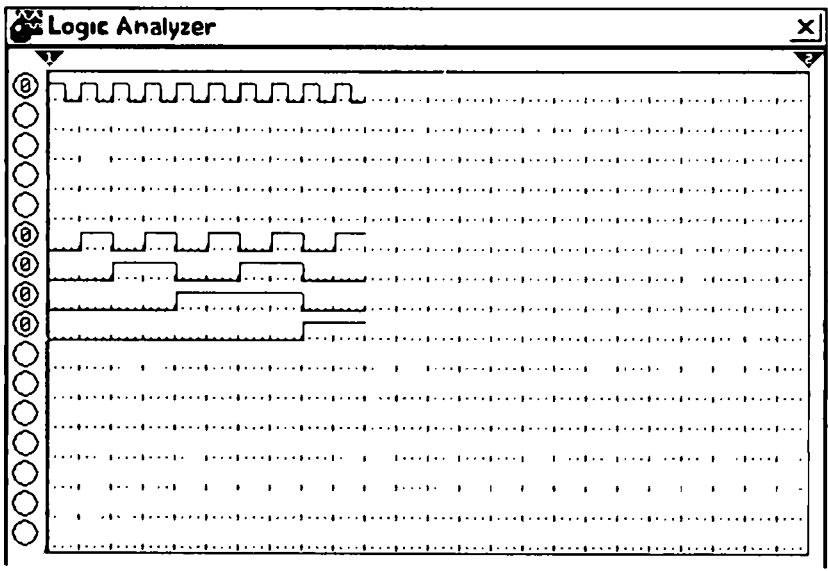
Часто работу счетчика требуется ограничить счетом до вполне определенного числа. Например, требуется, чтобы последней цифрой, выводимой на дисплей, была цифра 9. Такой счетчик должен считать от 0 до 9, всего 10 импульсов (считая с нулевого). Следовательно, это счетчик по модулю 10 или двоично-десятичный счетчик Он должен иметь, как минимум, четыре двоичных разряда (иначе нельзя сосчитать число $9_{10}=1001_2$). Очевидно, что для выполнения поставленных условий достаточно предусмотреть сброс счетчика в нуль импульсом, идущим после 9, т.е при $A_{16}=1010_2$. Отсюда видно, что на входы сброса R всех триггеров надо подать лог 1 после прихода цифры 9. Эта операция легко реализуется автоматически за счет организации обратной связи с выходов счетчика на входы сброса. Для того чтобы сброс происходил на числе 1010, поставим ЛЭ DD5 типа NAND, подключив его к выходам D и B (там, где появляются единицы), а выход – подключим к линии сброса R (см рис. 163,а). Этот прием «управляемого сброса» можно использовать для организации работы счетчиков по произвольному модулю.

Применяя принцип логической обратной связи к вычитающему счетчику, можно построить самоостанавливающийся счетчик например, для того, чтобы закончить счет на цифре 0. Для этого на входы J и K первого триггера, управляющие его переключением и запускающие далее за счет переносных связей переключением всей цепочки, надо подать лог 0 при появлении на счетчике сигнала 0000 Это нетрудно выполнить, собрав сигналы с выходов всех разрядов ЛЭ DD5 типа OR и подав выход с него на входы J и K первого триггера (см. рис 164,а)

Работа счетчиков в режимах сброса и остановки хорошо видна на графиках логического анализатора (см рис 163,б и 164,б)

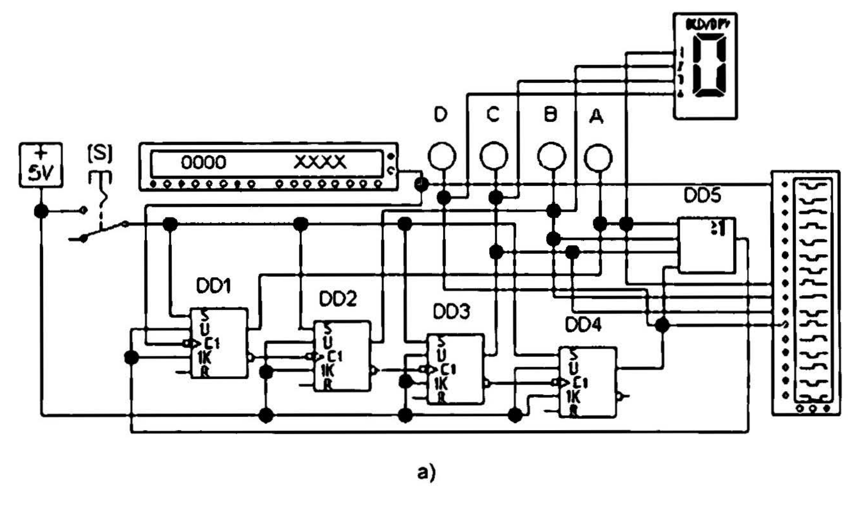


а)

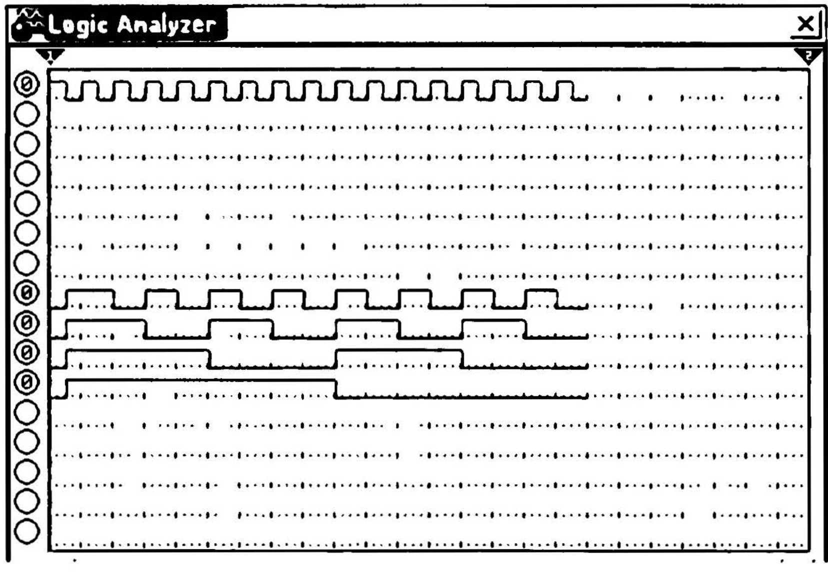


б)

Рис 163 Счетчик по модулю 10 (**EWB**)



а)



б)

Рис. 164. Самоостанавливающийся счетчик (**EWB**)

Синхронные счетчики

В счетчиках с последовательным переносом быстродействие ограничено временными задержками перенесения информации внутри счетчика. При параллельной загрузке информация подается одновременно на все триггеры счетчика синхронно по команде, поступающей с тактовым импульсом. Поэтому данный тип счетчиков состоит из синхронных триггеров, и задержка переключения всего счетчика определяется задержкой одного триггера. а не всей цепочки. В схемотехническом отношении цепочка триггеров в структуре счетчика остается для передачи информационных сигналов, но в дополнение к ней еще появляется комбинационная логика

В схемах синхронных счетчиков синхронизирующие входы соединяются параллельно и на них одновременно подается последовательность тактовых импульсов (см. рис. 165,а и 166,а). Выход Q переноса с триггера DD1 поступает на триггер DD2. Перенос в следующий разряд на триггер DD3 формируется ЛЭ DD5 NAND. Далее аналогично – на триггер DD4 – ЛЭ DD6 NAND. Результат работы счетчиков показан на графиках, представленных на рис 165,б и 166,б

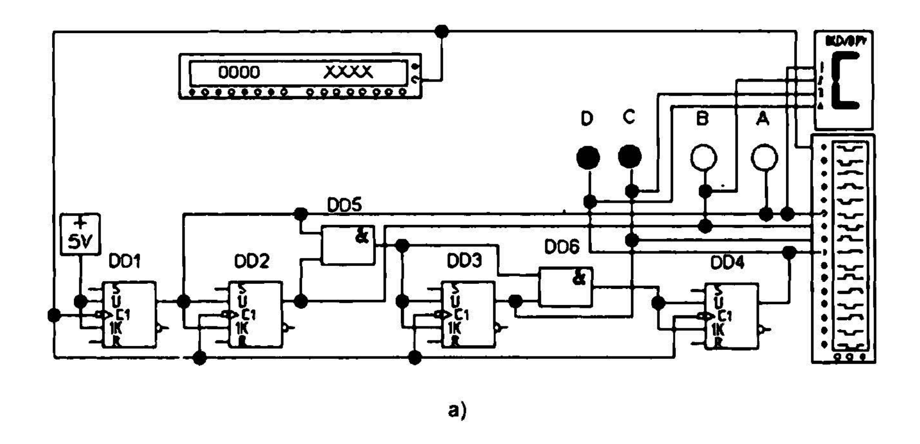
Схема синхронного двоично-десятичного счетчика показана на рис. 167,а

Для управляемого сброса счетчика при приходе сигнала, соответствующего числу 1010, здесь использован ЛЭ DD7 типа 4-Input NAND, два входа которого соединены с прямыми выходами триггеров DD1 и DD4, а два других – с инверсными выходами DD2 и DD3 Картина счета показана на экране логического анализатора (см рис 167,б)

Разновидностью синхронных счетчиков являются кольцевые счетчики, выполняемые на базе регистров сдвига. В простейшем кольцевом счетчике прямой выход регистра замыкают на его вход.

Для четырехразрядного регистра-счетчика кодовая единица, поданная на вход в первом такте, пройдя все разряды счетчика, спустя четыре такта снова попадет на его вход, и далее процесс будет повторяться.

Подобные устройства широко используются для определенного подключения входов программно-временных устройств или памяти

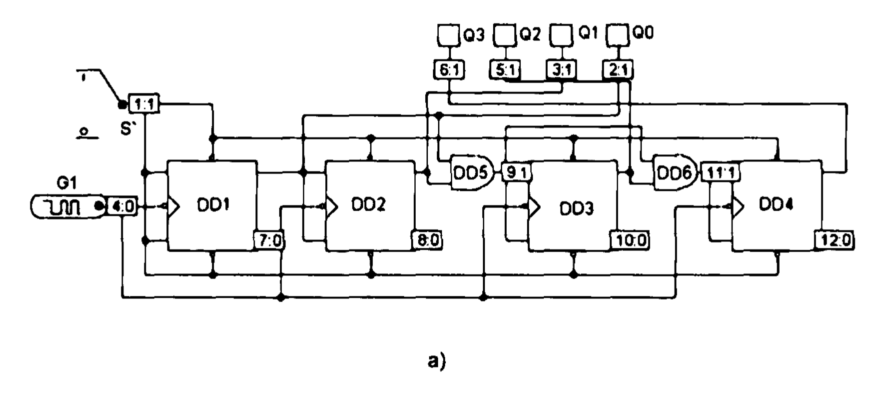


а)

Logic Analyzer

б)

Рис. 165. Синхронный четырехразрядный счетчик (**EWB**)



а)

DCOUNTER4.CIR

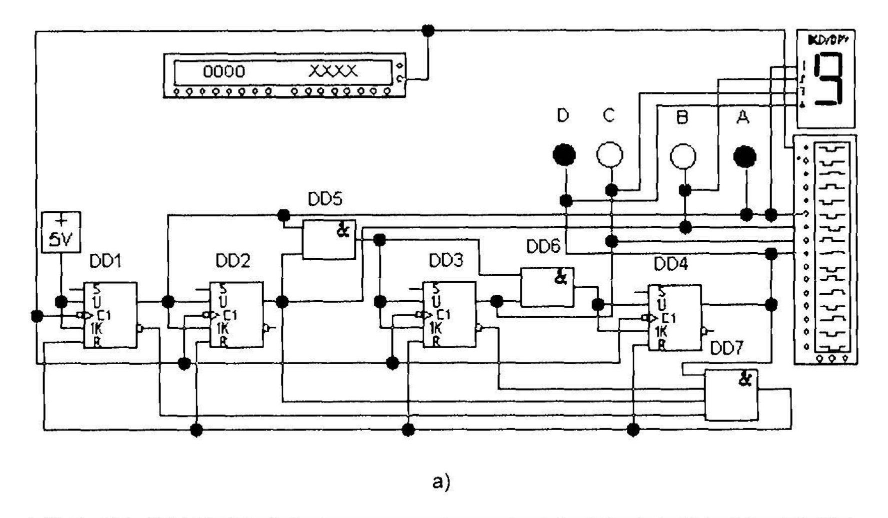
d(4)
d(2)
d(3)
d(5)
d(6)

0.000 3.000 6.000 9.000 12.000 15.000

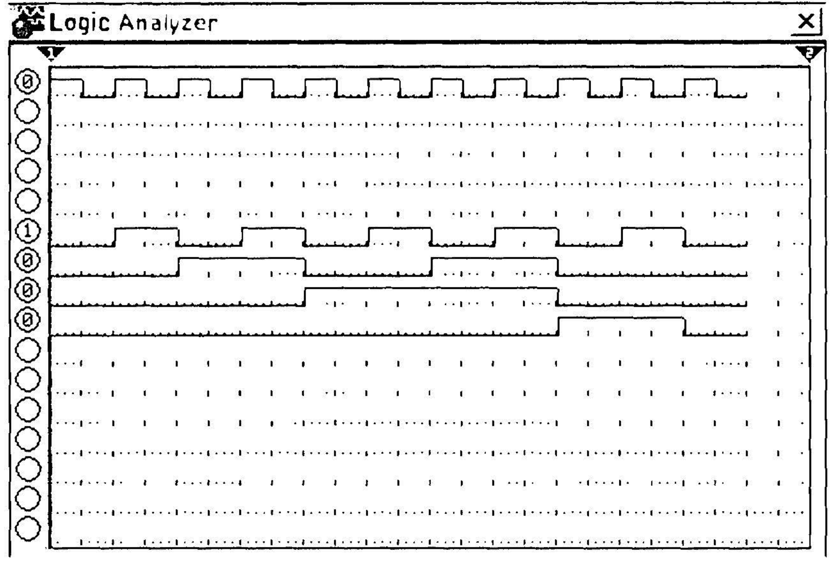
T

б)

Рис. 166 Синхронный четырехразрядный счетчик (МС)



а)



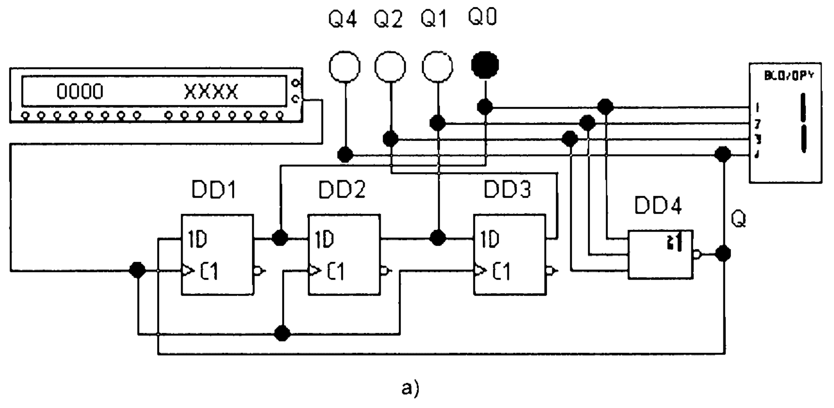
б)

Рис 167 Синхронный двоично-десятичный счетчик (**EWB**)

а)

б)

Рис. 168. Простейший кольцевой счетчик (**EWB**)



а)

Рис 169 (начало)

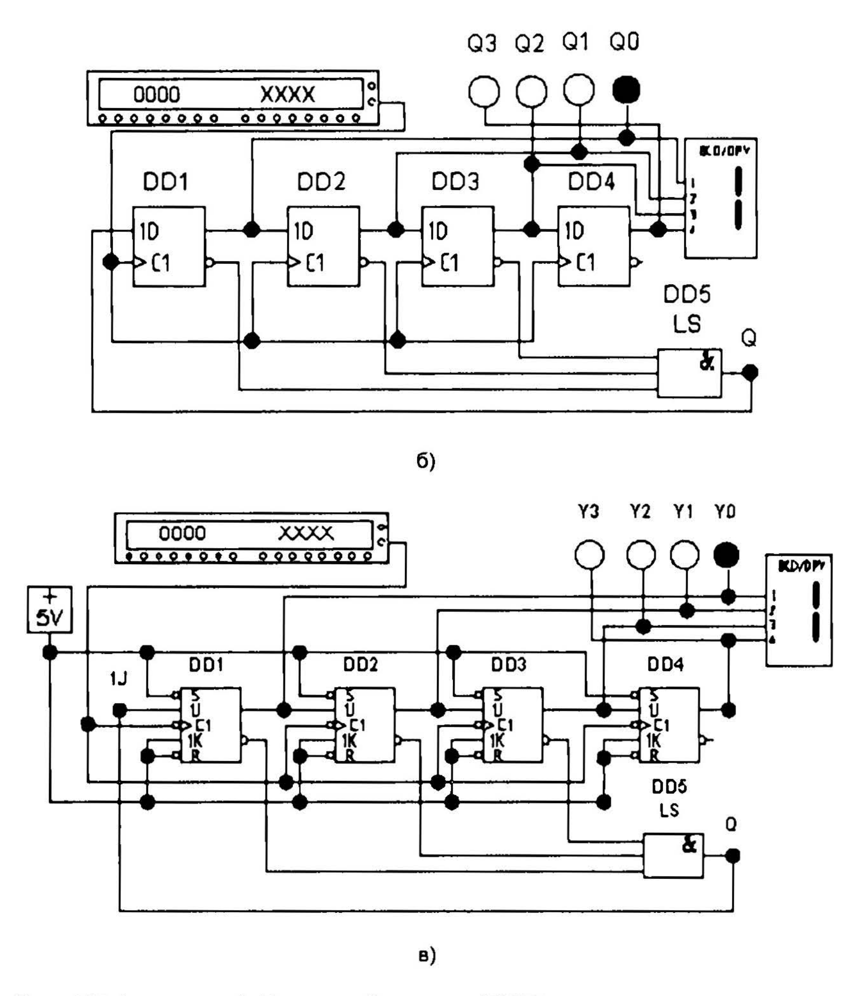


Рис. 169 (окончание). Кольцевой счетчик (**EWB**)

Схема простейшего кольцевого счетчика (рис. 168,а) аналогична схеме, показанной на рис. 146. В данном случае применены JK-триггеры с прямыми установочными входами R,S (часть которых здесь используется для начальной установки счетчика) Клавиша [Space] используется для начальной загрузки 0001 при начале счета Далее, с приходом импульсов, счетчик будет функционировать в кодовой последовательности при переводе в де-

сятичные числа следующим образом: 1, 2, 4, 8, 1, 2, 4, 8 . Единица будет двигаться от младших разрядов к старшим и, дойдя до самого старшего, начинать весь цикл сначала, пока не прекратится подача тактовых импульсов. Следовательно, этот счетчик имеет модуль, равный 4. На рис. 168,б показана та же схема, реализованная на D-триггерах. Здесь клавишей [Space] вначале устанавливается произвольный код (в данном случае 1001), затем последует 2, 4, 8, и далее будет повторяться кодовая последовательность 1, 2, 4, 8. Недостатками подобных простейших схем при их практическом применении являются сбои, вызванные появлением лишних или исчезновением нужных кодовых единиц в кольце. От лишних единиц можно избавиться, введя дополнительную логическую цепь, разрешающую передачу единицы из последнего триггера в первый только при наличии нулей в остальных разрядах В схеме на рис. 169,а для этой цели использован ЛЭ DD4 типа 3*ИЛИ-НЕ*, выходной сигнал с которого подается на вход 1D триггера DD1. Этот же логический элемент здесь выполняет одновременно роль последнего счетного разряда Обычно в схемах кольцевых счетчиков, имеющих минимальное число триггеров, добавляют еще один триггер, для сдвига последнего разряда. Соответствующие схемы показаны на рис. 169,б,в, причем в них использован ЛЭ DD5 3*И* типа LS. Подобные ТТЛ микросхемы Шоттки (S) имеют малое энергопотребление (L – Low – низкий) и время задержки порядка 10 нс.

Интересной разновидностью схем кольцевых счетчиков, построенных на основе регистра сдвига, является их замыкание перекрестной обратной связью В этом случае сигналом обратной связи, который берется с последнего триггера в качестве выходного, является не прямой, а инверсный сигнал, подаваемый на счетный вход первого триггера.

Подобные счетчики называют «счетчиками Джонсона». Счетчик Джонсона на D-триггерах показан на рис. 170,а. Последовательность состояний счетчика соответствует ряду десятичных чисел: 1, 3, 7, 15, 14, 12 , 8, 0, 1, 3, 7, 15, 14, 12 , 8, 0... За один цикл счетчик проходит 8 состояний, т.е. это счетчик по модулю 8, что в два раза больше, по сравнению с аналогичным счетчиком, но с прямой обратной связью. Первые четыре такта счетчик заполняется волной единиц, а вторые четыре такта – очищается волной нулей. Счетчик на JK-триггерах, показанный на рис. 170,б, в схемотехническом отношении выглядит еще изящнее: с обоих его выходов сигналы идут перекрестно обратно на входы (Q→1K и Q`→1J). Не случайно, поэтому подобные счетчики называли

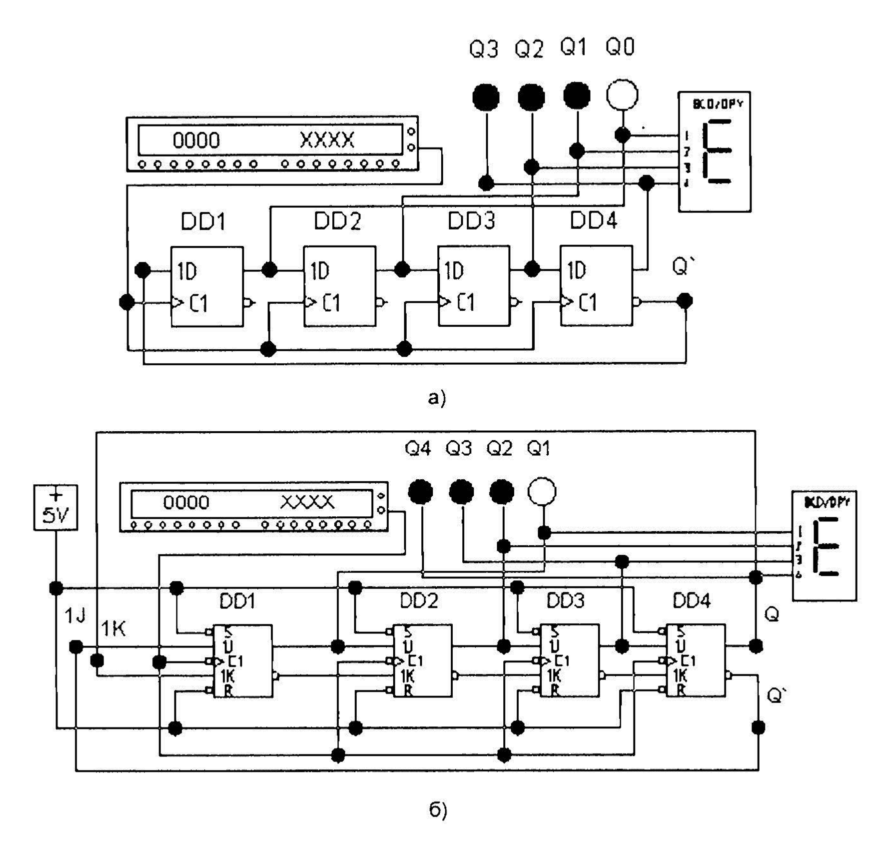


Рис. 170. Счетчик Джонсона (**EWB**)

также счетчиками Мёбиуса, проводя аналогию с известной в математике лентой (листом) Мёбиуса (по фамилии немецкого геометра XIX в.). В топологии (раздел геометрии) лента Мёбиуса служит примером односторонней поверхности. Действительно, если взять ленту (прямоугольник KJQQ`) и совместить (склеить) ее противоположные края (совмещая точки K с Q, а J с Q`), то образуется обычное кольцо, имеющее две поверхности: вешнюю и внутреннюю. Двигаясь вдоль средней линии этой замкнутой ленты, мы никогда не попадем с наружной стороны на внутреннюю, и наоборот. Пространственный период обращения здесь равен исходной длине ленты. Теперь, взяв ту же ленту, при совмещении противоположных сторон перевернем ее половину на

180° (совместим точки K с Q`, а J с Q) Обходя в выбранном направлении вдоль этого своеобразного кольца, мы с неизбежностью пройдем по обеим его сторонам, и период будет равен удвоенной начальной длине ленты. Вернемся к электронике и посмотрим геометрически на работу кольцевого счетчика на рис 146 Это логическое электронное кольцо сделано из «триггерной ленты» так, что вход напрямую связан с выходом (Q→1J и Q`→1K). Модуль счета равен числу триггеров (M=4). В счетчике же Джонсона, показанном на рис. 170,б, связи выходов и входов «триггерной ленты» перекрестные, а именно: Q→1K и Q`→1J В результате и модуль счета становится в два раза больше (M=8). Действительно, все это очень даже напоминает ленту Мебиуса.

Счетчики на микросхемах

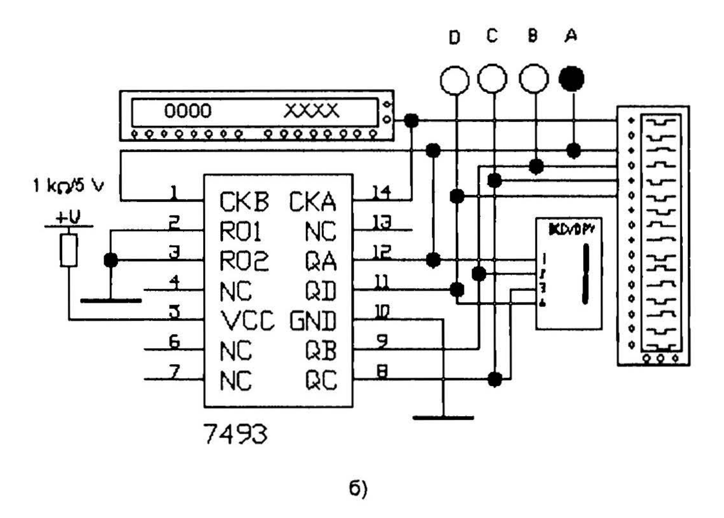
Выпускается несколько десятков типов микросхем самых разнообразных счетчиков как в серии ТТЛ, так и в серии КМОП. В качестве примера составим схемы моделей для некоторых из них.

В программе **EWB** войдем в меню Digital, откуда буксируем на рабочее поле пиктограмму с изображением корпуса микросхемы CNT Σ (CNT от англ. COUNTERS – счетчики). В открывшейся библиотеке моделей выберем микросхему ТТЛ 7493 Согласно таблице состояний счетчика, полученной из предметной помощи программы (см. рис. 171,а), он четырехразрядный (по числу двоичных разрядов), считающий в двоичном коде от 0 до 15 (M=16). Микросхема состоит из четырех триггеров, внутренне связанных

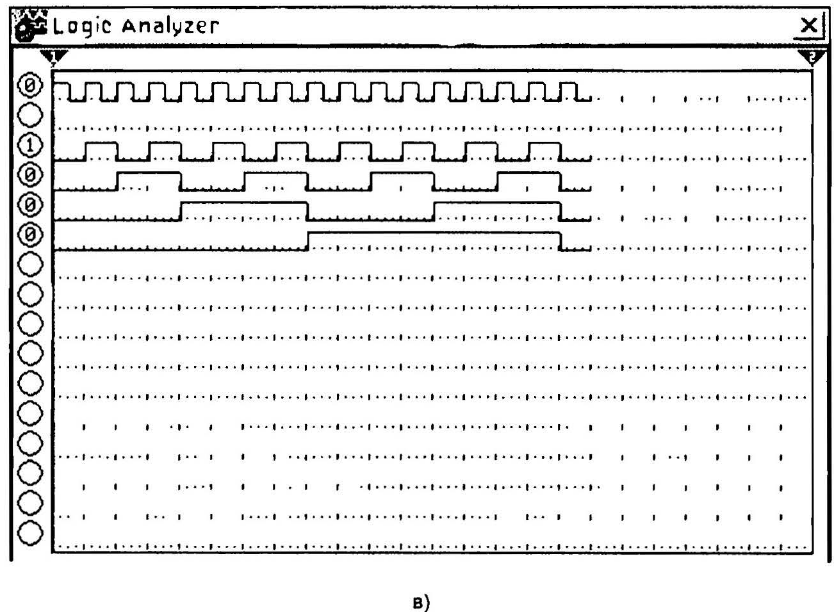
7493 (4-bit Binary Counter) The 7493 counts from 0 to 15 in binary		
Reset in		Output
RO1	RO2	Qd Qc Qb Qa
1	1	0 0 0 0
0	X	Count
X	0	Count

а)

Рис. 171 (начало)



б)



в)

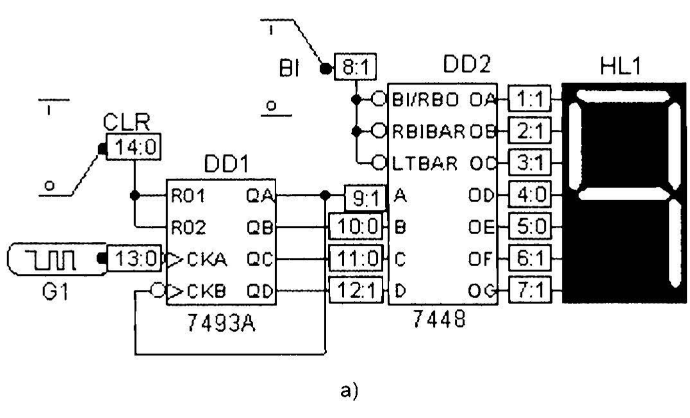
Рис. 171 (окончание). Синхронный четырехразрядный счетчик на микросхеме 7493 (**EWB**)

между собой так, что образуются два счетчика-делителя: один на два и другой – на восемь. Все триггеры имеют общий сброс (верхняя строка состояний в приведенной таблице), который осуществляется подачей высокого уровня на выводы RO1(ножка 2 микросхемы) и RO2 (ножка 3). В нормальном режиме работы хотя бы на один из этих входов должно подаваться напряжение низкого уровня; в этой таблице не показано, но счетный режим будет и при RO1= RO2=0. Поскольку первый триггер не связан с тремя другими, то возможны различные варианты применения микросхемы. Соберем стандартный синхронный суммирующий счетчик с M=16. Для этого надо подать стандартное питание, соединить между собой синхронизирующие выводы QA и CKB и подать счетные импульсы на вход CKA, соответственно, индикация проводится на выходах QA...QD (см. рис. 171,б). Напоминаем, что буквами NC (Not Connection) обозначены холостые выводы микросхемы В качестве источника постоянного напряжения к входу VCC, здесь и далее применительно к микросхемам, будет использоваться источник с напряжением 5 В, последовательно с которым включен резистор 1кОм (так называемый Pull-Up Resistor – дословно – вытягивающий, а по смыслу – балластный резистор). Величина напряжения источника и сопротивление резистора могут быть установлены другими путем редактирования в окне свойств. Нетрудно видеть, что приведенные графики (рис. 171,в) совпадают с полученными ранее для подобного же счетчика, но собранного из отдельных компонентов (см. рис. 165,б). Аналогично работает и модель счетчика на микросхеме 7493А в программе **MC** (см. рис. 172). Еще раз обратите внимание на необходимость внешней перемычки между выходом QA и входом CKB (см рис. 171,в и 172,б)

Теперь познакомимся с более универсальным, а значит и более сложным интегральным счетчиком 74192. Как только что было описано выше (применительно к микросхеме 7493), выберем его из библиотеки моделей компонентов программы **EWB** и изучим таблицу его состояний (см. рис 173,а).

Это синхронный (Sync) реверсивный (Up/Down) двоично-десятичный (BCD) счетчик (Counter), имеющий вход сброса (Clear) и отдельные входы для счета импульсов в прямом (Up) и обратном (Down) направлениях. В режиме счета на вход загрузки данных (LOAD)` подается напряжение высокого, а вход сброса CLEAR – низкого уровня (см. две нижние строки в таблице на рис. 173,а). Значение, хранящееся в счетчике, увеличивается на 1 при приходе фронта импульса (POS) для счета в прямом направле-

нии (Count Up) и уменьшается на 1 соответственно при обратном счете (Count Down). В этих случаях на один из входов должно подаваться напряжение высокого уровня.



а)

7493.CIR

d(13)
d(9)
d(10)
d(11)
d(12)

0.0u 1.0u 2.0u 3.0u 4.0u 5.0u

T

б)

Рис. 172. Синхронный четырехразрядный счетчик на микросхеме 7493А (**МС**)

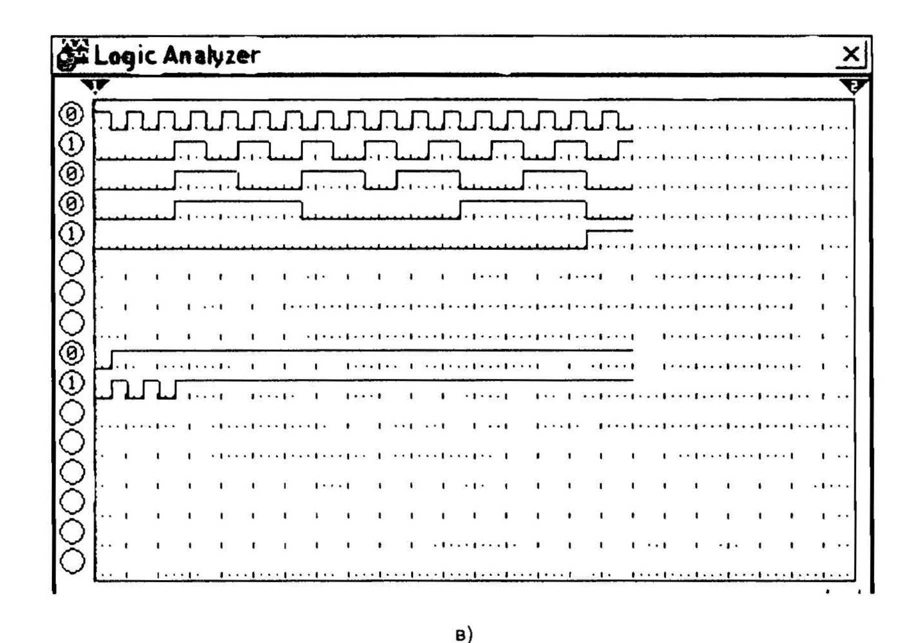
При программировании счетчика необходимые данные в двоично-десятичном коде подаются на входы A...D при кратковременной подаче низкого уровня напряжения на вход (LOAD)`. Собрав соответствующую схему (см. рис. 173,б), предусмотрим в ней возможности указанных режимов работы за счет программирования установкой ключей: C – очистка, U – направление счета, T – тактовые импульсы, L – загрузка.

74192 (Sync BCD up/down Counter) Up/Down Counter truth table:						
CLEAR	UP	DOWN	(LOAD)`	PARALLEL A B C D	OUTPUTS QA QB QC QD	(CO)` (BO)`
1	X	X	X	X X X X	0 0 0 0	1 1
0	X	X	0	X X X X	A B C D	1 1
0	POS	1	1	X X X X	Count UP	* *
0	1	POS	1	X X X X	Count Down	* *

а)

б)

Рис. 173 (начало)



в)

Рис. 173 (окончание) Синхронный реверсивный программируемый четырехразрядный счетчик на микросхеме 74192 (**EWB**)

Программирование будем проводить по таблице:

№	Режим счетчика	Установки ключей			
		C	U	T	L
1	Очистка	1	X	X	X
2	Загрузка	0	X	X	0
3	Обратный счет	0	UP	DOWN	1
4	Прямой счет	0	DOWN	UP	1

Работа счетчика в соответствии с этой таблицей показана на рис 173,в На графиках видно, что сначала произошла очистка счетчика. Затем загрузка 0111 от генератора слов. На этом такте ключ L должен находиться в нижнем положении и тогда 0 «придет» к нему от генератора слов Далее происходит обратный счет

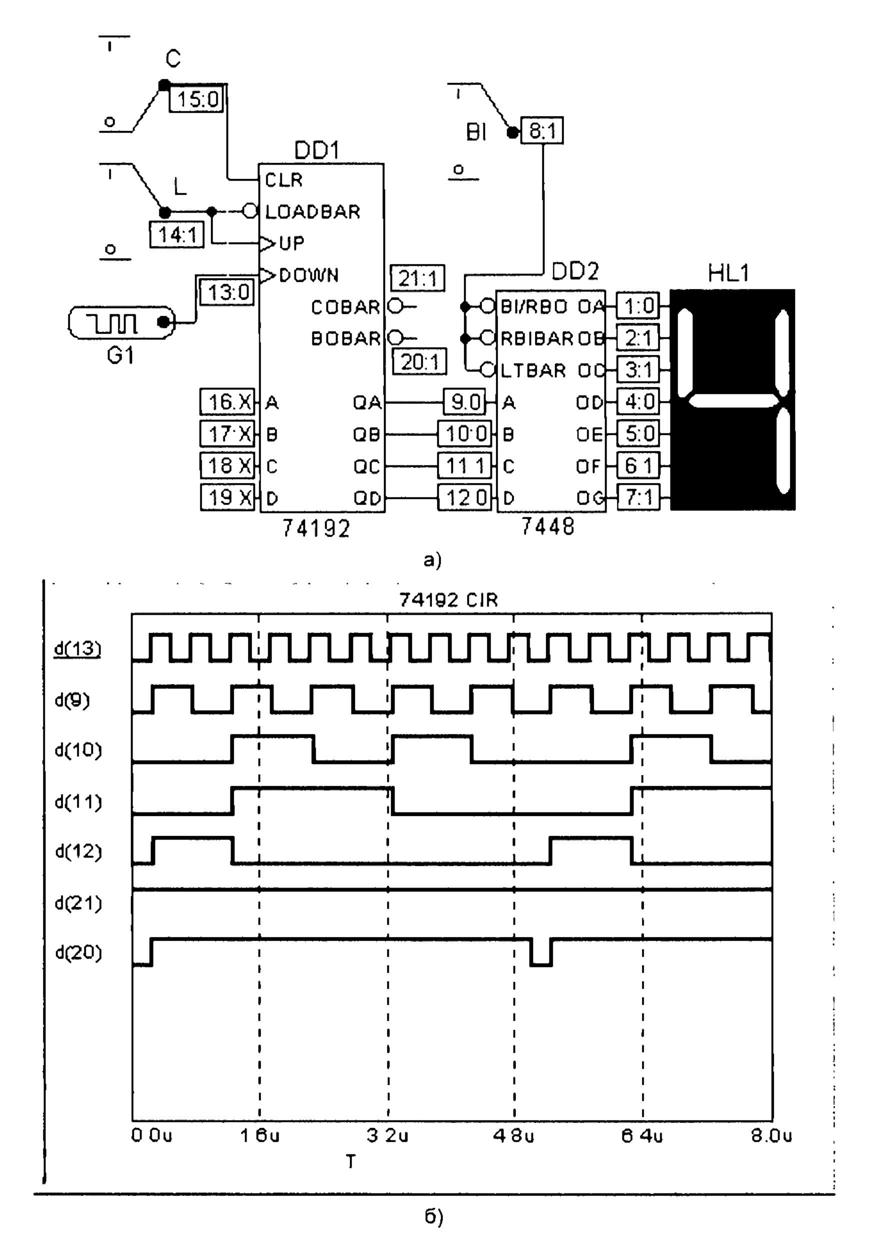


Рис 174. Синхронный реверсивный программируемый четырехразрядный счетчик на микросхеме 74192 (**МС**)

до 0001 и затем прямой счет до 1001. На двух нижних графиках показаны сигналы, снимаемые с выходов CO` и BO`, которые могут использоваться при каскадировании счетчиков. Это выходы переноса переполнения при прямом счете и заёма – при обратном.

Рассмотрим работу этой же микросхемы в программе **MC** В данном случае, чтобы продемонстрировать на одном графике все случаи, пришлось бы вместо ключей поставить генераторы цифровых сигналов и специальным образом задать согласованный режим их работы. Поэтому ограничимся наблюдением режима обратного счета. Схема и результаты показаны на рис 174,а,б.

4. ЦИФРОВЫЕ УСТРОЙСТВА

4.1. Арифметическо-логические устройства

Друг мой! Знаешь ты уже
Вычитанье и сложе-,
Умноже-нье и деленье
Просто всем на удивленье.
Так дерзай! Пусть славы эхо
О твоих гремит успехах,
Станешь ты, хоть скромен вид
Знаменитей чем Эвклид!
Льюис Кэррол.
История с узелками

Арифметическо-логическое устройство – АЛУ, на английском языке, соответственно, Arithmetic-Logic Unit – ALU, представляет собой сложную цифровую микросхему, которая может выполнять все комбинационные функции, а также выполнять ряд других операций. Не случайно поэтому, большая часть информации в компьютерах обрабатывается именно в АЛУ, которое входит составной частью в специальную интегральную схему, именуемую центральным процессором (Central Processor Unit – CPU). В первоначальном компьютере IBM PC микросхема CPU была изготовлена Intel Corporation и названа 8080 Центральный процессор собственно и определяет тип ПК: например, при 486-й CPU и машину называют 486-я («четверка»), при CPU типа Pentium и ПК называют Pentium (от греч Pente – пять). Если уподобить этот микропроцессор голове, то АЛУ – это думающая часть ее мозга (другими частями являются память и устройство управления).

АЛУ выполняет одну из главных функций микропроцессора – обработку данных Обычно АЛУ имеет два входных порта и один выходной, служащих, соответственно, для ввода и вывода слов Входные порты снабжаются буферными регистрами для хранения одного слова данных. Последние здесь не рассматриваются, как, впрочем, и остальные, сопряженные с АЛУ: аккумулятор, регистр состояний и шины Они вместе с АЛУ будут рассмотрены в последнем разделе, посвященном примеру модели простейшей ЭВМ Перечень функций собственно АЛУ зависит от архитектуры микропроцессора и различен для машин разных типов

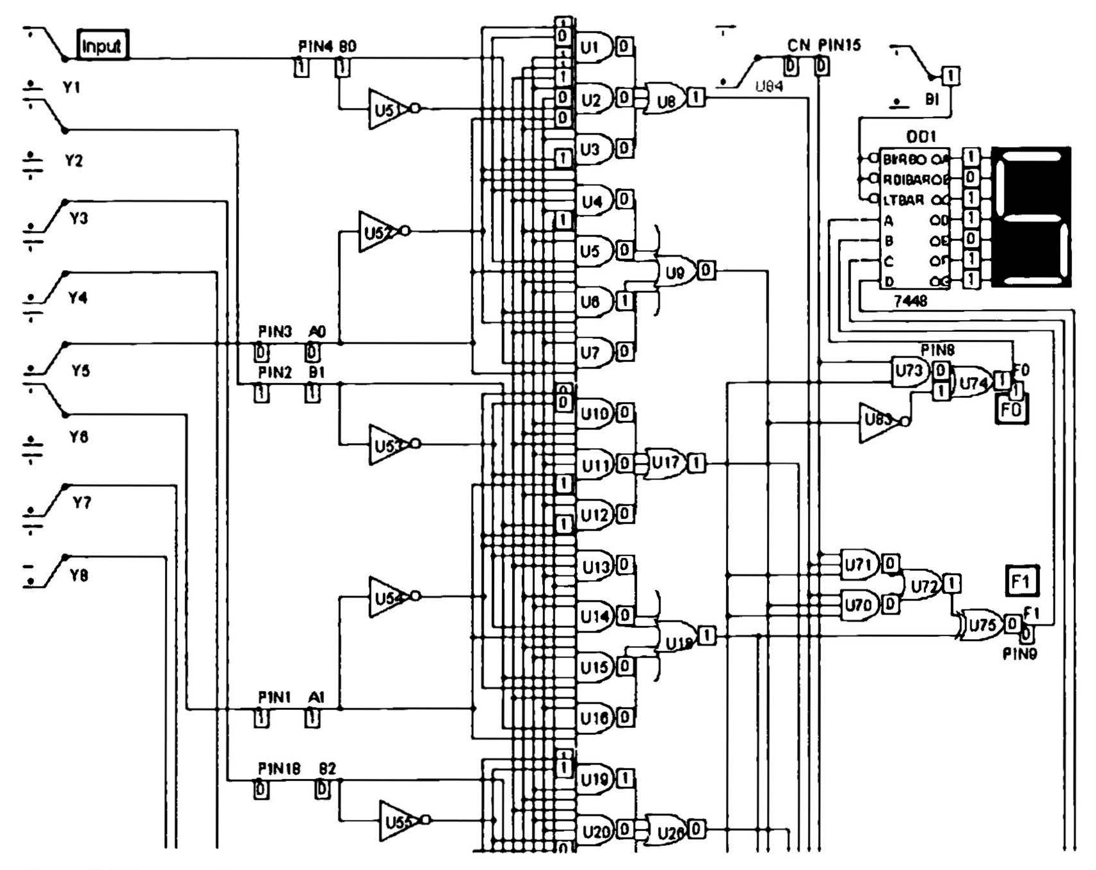


Рис 175 (начало)

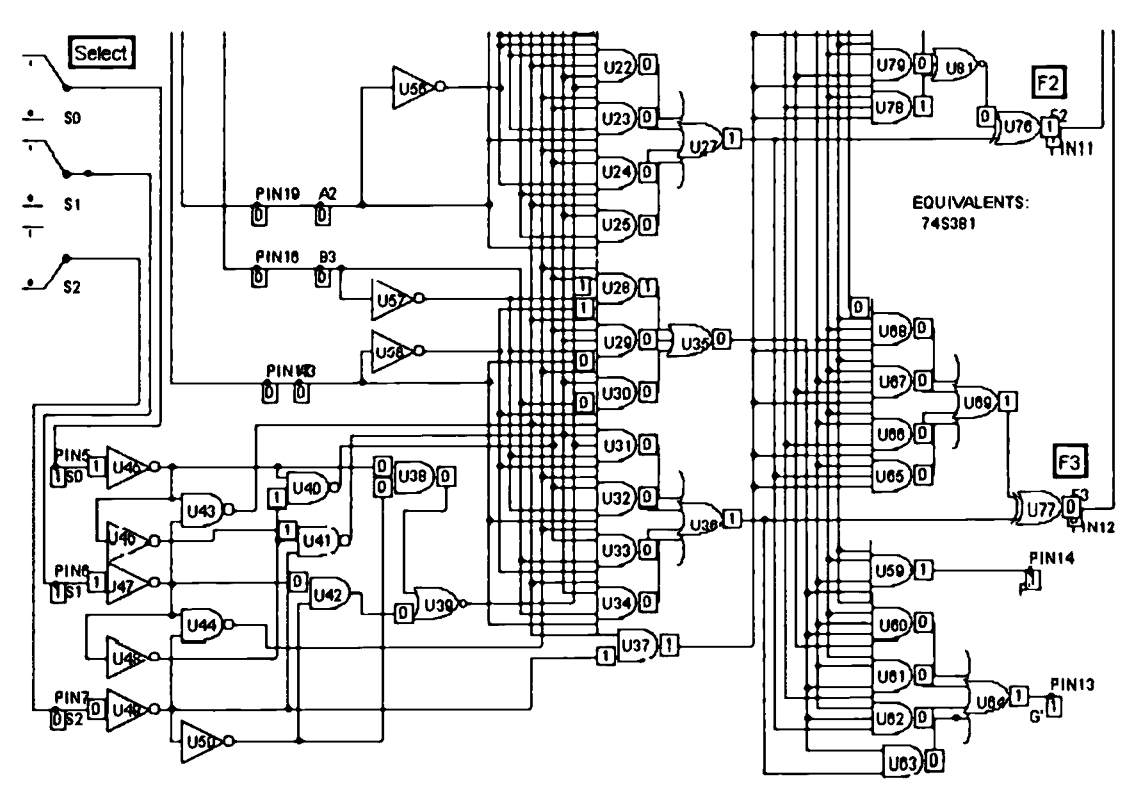


Рис. 175 (окончание) Логическая структура четырехразрядного АЛУ на микросхеме 74381 (МС)

АЛУ 74381

Рассмотрим АЛУ на микросхеме 74381, с помощью которого можно выполнять три арифметические и три логические операции над четырехразрядными операндами. Логическая структура этого АЛУ представлена на рис. 175 Показанная схема взята из библиотечного файла программы **МС**, который был частично отредактирован и дополнен компонентами, необходимыми для исследований поведения модели

В общих чертах работа данной схемы происходит следующим образом. Операнды A0–A3 и B0–B3 поступают побитно на соответствующие входы Input. В модели на рис 175 для этой цели использованы цифровые ключи Y1–Y8. Таблица истинности АЛУ показана на рис. 176, в верхней половине которой собраны арифметические, а в нижней – логические операции.

Код операции			Арифметическо-логическая операция
S2	S1	S0	
L	L	L	Сброс
L	L	H	В минус А
L	H	L	А минус В
L	H	H	А плюс В
H	L	L	А «+» В
H	L	H	А + В
H	H	L	АВ
H	H	H	Установка

Рис. 176. Таблица истинности четырехразрядного АЛУ 74381

Совет При чтении логических и арифметических выражений не путайте логическую операцию *ИЛИ*, обозначаемую здесь знаком **+**, с арифметическим сложением, имеющим такое же обозначение, а здесь выделяемое словом «плюс» (plus) Аналогичная проблема может возникнуть с логической операцией *И*, которую можно перепутать с

арифметическим умножением. Поэтому обращайте внимание на контекст. *Исключающее ИЛИ* (XOR) здесь обозначено знаком плюс, взятым в кавычки «+». Логические уровни обозначены: низкий – L (от Low), высокий – H (от High). Последнее не примите за русское Н (от Низкий), также используемое в описаниях.

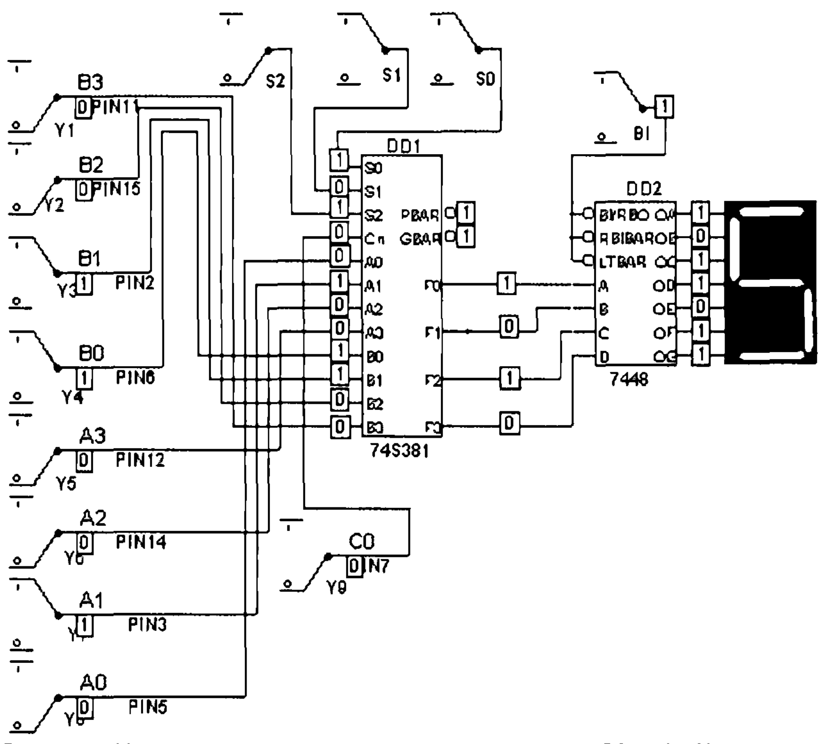


Рис 177. Четырехразрядное АЛУ на микросхеме 74S381 (**МС**)

Согласно данной таблице набором S2, S1, S0 выбирается (Select) код операции для выполнения необходимой функции работы АЛУ, результат которой снимается с выводов F0–F3. Очистка АЛУ производится подачей низкого уровня на входы S0–S2, а при подаче на эти входы высокого уровня на всех выходах F устанавливается лог. 1. В приведенной модели в первом случае после анализа Transient на буквенно-цифровом индикаторе заго-

рится 0, а во втором – на нем не будет никакого изображения, что соответствует $(1111)_2 = F_{16}$ (см. ранее рис. 101,в). На приведенном рис. 175 ключи выбора имеют следующую установку: S0=1, S1=1, S2=0, что в соответствии с ТИ дает операцию арифметического сложения. Входные операнды, выбранные ключами Y1–Y8, таковы. $A=(0010)_2=2_{10}$ и $B=(0011)_2=3_{10}$. Их поразрядное сложение с учетом переноса дает $C=(0101)_2 = 5_{10} = 5_{16}$. Именно это число загорается на выходном дисплее (см. рис. 175). Поскольку программа одновременно показывает и состояния всех цифровых узлов, то по ним можно просмотреть всю логику совершаемых операций, что весьма удобно при наладке схем. Работа аналогичного АЛУ «запакованного» в корпус, показана на рис. 177.

АЛУ 74181

Рассмотрим АЛУ на микросхеме средней интеграции 74181, которое является параллельным четырехразрядным прибором, способным выполнять 16 арифметических и все 16 возможных логических операций над двумя параллельными четырехразрядными словами-операндами A0–A3 и B0–B3. Важными арифметическими операциями являются сложение, вычитание, передача данных, дифференцирование, положительное приращение, отрицательное приращение, инвертирование и удвоение. Система обозначений, принятых в данном АЛУ, такова:

A0–A3, B0–B3 – входы операндов (активных низких);
S0, S1, S2, S3 – функции выбора режимов;
M – вход контроля режима;
C0 – вход переноса;
F0, F1, F2, F3 – выходы функций (активных низких);
A=B – выход компаратора;
G – выход формирования переноса (активный низкий);
P – выход распространения переноса (активный низкий);
C4 – выход переноса.

Необходимая операция выбирается по четырем линиям выборки S0...S3 и линии управления режимом M, которая имеет низкий уровень для арифметических операций и высокий для логических операций. Это непосредственно следует из рассмотрения ТИ, заимствованной из раздела Help программы **EWB** и показанной на рис. 178.

74181 (Alu/Function Generator)
ALU/Function Generator truth table. Active-low Data

Selection S3 S2 S1 S0	M=H Logic Functions	M=L; Arithmetic Operations Cn=L (no carry)	Cn=H (with carry)
0 0 0 0	F = A`	F=A MINUS 1	F=A
0 0 0 1	F= (AB)`	F= AB MINUS 1	F=AB
0 0 1 0	F= A`+B	F= AB` MINUS 1	F=AB`
0 0 1 1	F= 1	F=MINUS 1(2's comp)	F=Zero
0 1 0 0	F= (A+B)`	F=A PLUS (A+B`)	F=A PLUS(A+B`) PLUS 1
0 1 0 1	F= B`	F=AB PLUS(A+B`)	F=AB PLUS(A+B`) PLUS 1
0 1 1 0	F= (A "+" B)`	F=A MINUS B MINUS 1	F= A MINUS B
0 1 1 1	F= A+B`	F= A+B`	F= (A+B`) PLUS 1
1 0 0 0	F= A`B	F= A PLUS (A+B)	F=A PLUS(A+B)PLUS 1
1 0 0 1	F= A "+" B	F= A PLUS B	F=A PLUS B PLUS 1
1 0 1 0	F = B	F= AB PLUS (A+B)	F=AB` PLUS (A+B)PLUS 1
1 0 1 1	F = A + B	F = (A + B)	F=(A+B)PLUS 1
1 1 0 0	F = 0	F = A PLUS A	F=A PLUS A PLUS 1
1 1 0 1	F = AB`	F= AB PLUS A	F=AB PLUS A PLUS 1
1 1 1 0	F=AB	F=AB` PLUS A	F=AB PLUS A PLUS 1
1 1 1 1	F = A	F = A	F= A PLUS 1

Рис. 178. Таблица истинности четырехразрядного АЛУ на микросхеме 74181

Устройство имеет «вход переноса», «выход переноса» для сквозного переноса при каскадировании узлов и две вспомогательные функции ускоренного переноса («формирование переноса» и «распространение переноса») для использования ускоренного переноса микросхемы 74182. Микросхема обеспечивает выход условия A=B с открытым коллектором, который можно связать схемой *И* с выходами A=B других АЛУ для выделения условия всех высоких уровней выходов нескольких устройств.

В логической части АЛУ 74181(как и в выше рассмотренной микросхеме 74381, показанной на рис. 175) четыре идентичные логические схемы *И/ИЛИ* открывают входные операнды A и B четырьмя линиями выборки S0–S3 для формирования требуемых вспомогательных функций *И* и *ИЛИ* первого уровня Затем они используются для формирования функций суммы и переноса Внутренний ускоренный перенос обеспечивает высокое быстродействие. Выход A=B формируется путем восприятия всех единичных состояний на F-выходах. Когда управляющий вход M высокий, то распространение переносов запрещено и на выходах вырабатываются логические функции. Имеющиеся в АЛУ функции образуют закрытый набор, такой, что инверсия логических входов дает имеющуюся в этом наборе функцию. Поэтому это устройство выполняет те же логические и арифметические функции в активном высоком представлении, как и в активном низком представлении, но при отличном коде выборки. Микросхема допускает также смешанное кодирование состояний.

Представленная здесь микросхема (см. рис. 179) соответствует режиму сумматора без переноса. Значения четырехразрядных операндов A и B на входе задаются с помощью ключей [1] – [8] и в шестнадцатеричном коде отображаются одноименными алфавитно-цифровыми индикаторами. На выходах F0–F3 результат суммирования отображается индикатором C. Изменяя состояния сигналов на управляющих входах, можно промоделировать большинство функций АЛУ, используемых в микропроцессоре.

В качестве примера рассмотрим работу АЛУ для выполнения логической функции *Исключающее ИЛИ* (XOR), широко используемой в микропроцессоре для осуществления разнообразных проверок. В ТИ на рис. 178 эта операция для удобства набора с клавиатуры обозначена знаком плюс, взятым в кавычки «+». Поскольку эта логическая операция осуществляется при тех же положениях ключей, что и при рассмотренном выше арифметическом сложении, то оставим их без изменения за исключением M, который переведем на высокий уровень Для последующей

проверки результата выход инвертируем, создав специальный субблок 4×NOT (см. рис. 180,а,б).

Окончательно схема и результат ее работы показаны на рис 180,в Для проверки этого результата можно собрать простейшую схему на восьмивходовом БЛЭ типа XOR или подсчитать его алгебраически

F=AB`+A`B=(0010)(1100)+(1101)(0011)= 0001.

Такой результат получается при поразрядном выполнении соответствующих булевых логических операций над операндами и он же виден на выходном дисплее С (см. рис. 180,в).

Работа того же АЛУ в режиме сумматора показана на рис 181 в программе **МС** При арифметическом сложении двух «пятерок» получается шестнадцатеричная «десятка», имеющая на буквенно-цифровых дисплеях очень своеобразный вид

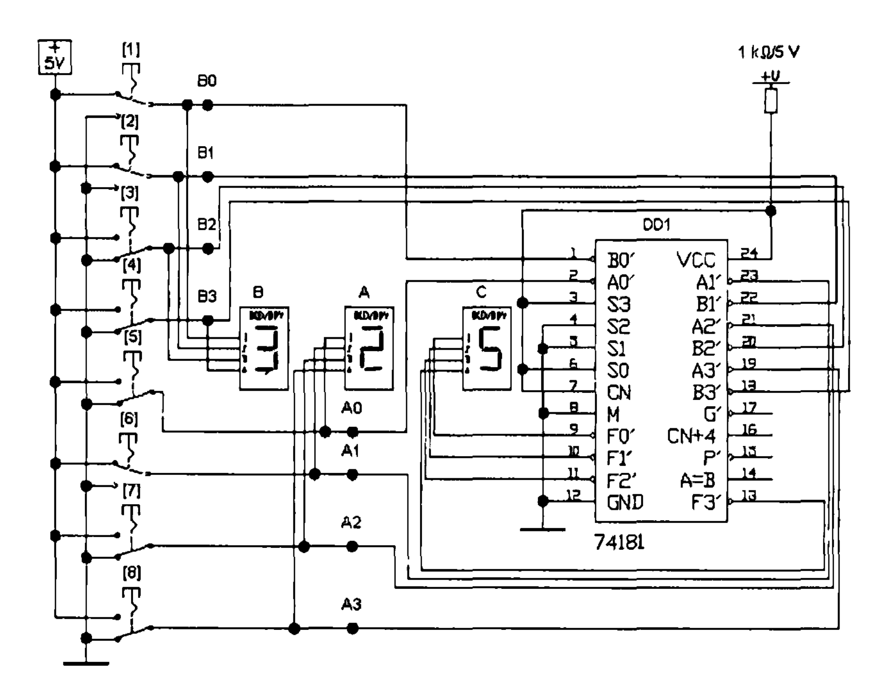


Рис. 179. Сумматор на четырехразрядном АЛУ 74181 (**EWB**)

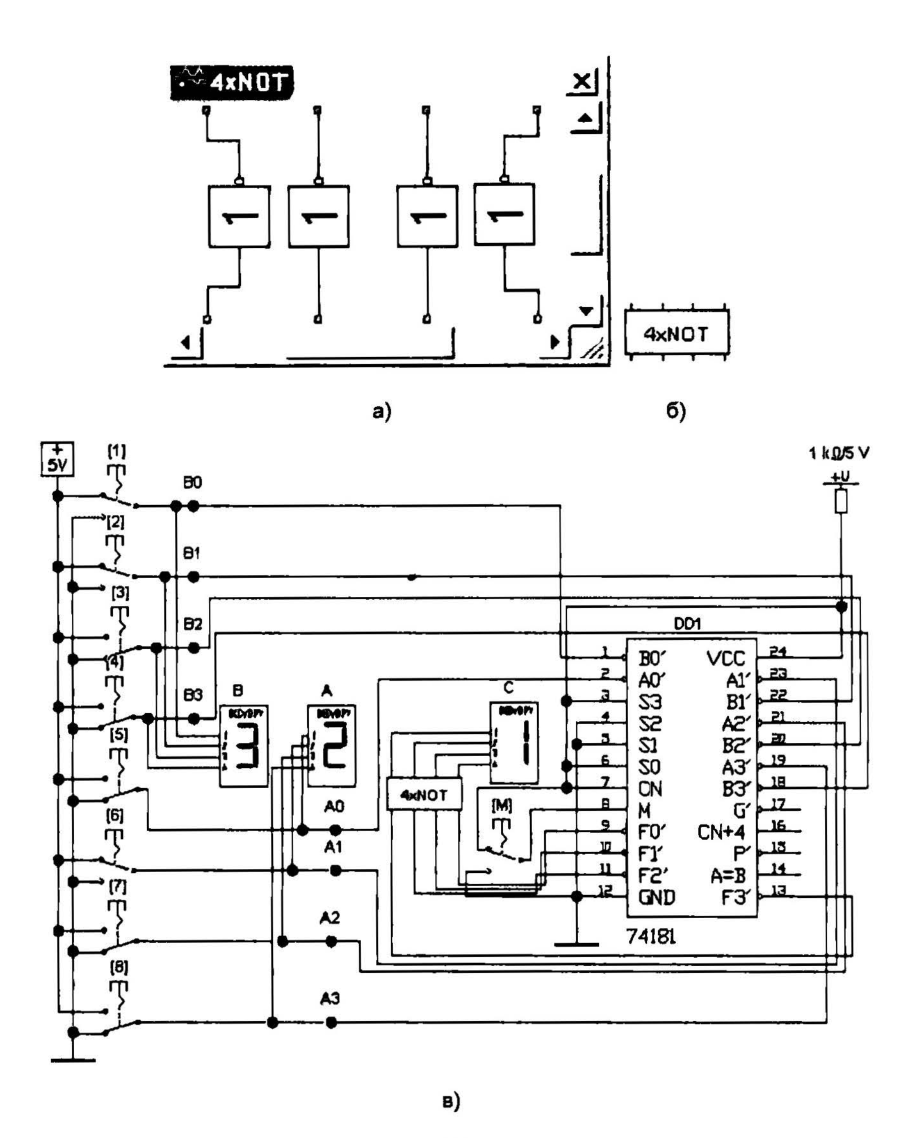


Рис. 180. Операция *Исключающее ИЛИ* на четырехразрядном АЛУ 74181 (**EWB**)

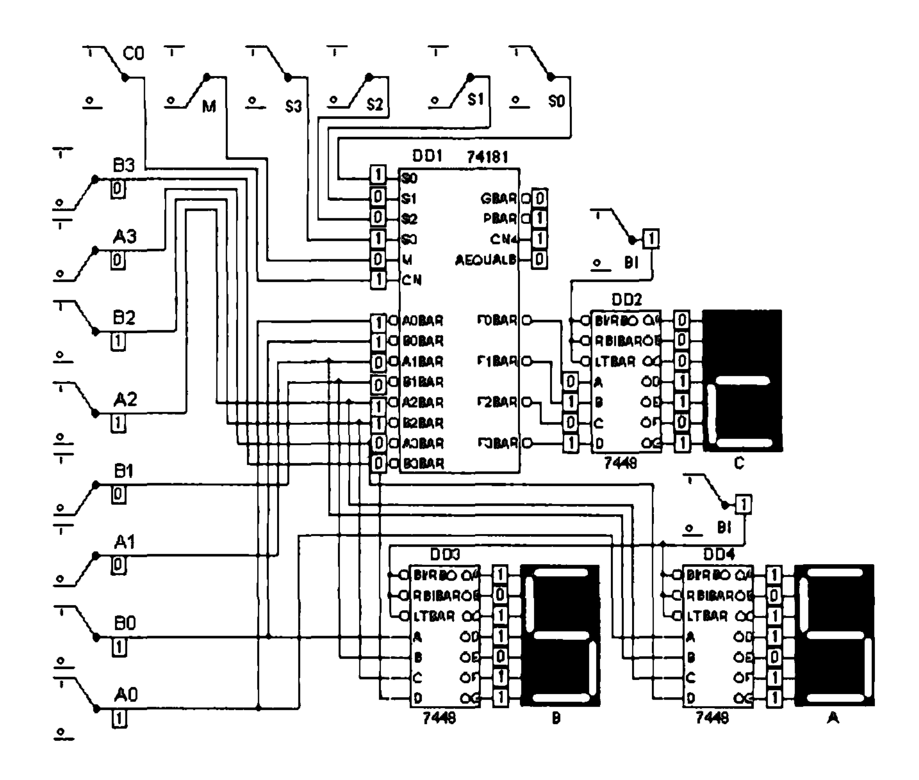


Рис. 181. Сумматор на четырехразрядном АЛУ 74181 (**MC**)

Как указывалось выше, микросхема 74181 дает также возможность осуществлять вспомогательные функции («формирование переноса» и «распространение переноса»), которые могут быть использованы вместе с микросхемой 74182 для выдачи общего ускоренного переноса или блока сквозного ускоренного переноса. В последнем режиме полное АЛУ разбито на шестнадцатиразрядные блоки, каждый из которых снабжен своим собственным ускоренным переносом, допускающим сквозной перенос между тактовыми сигналами. Микросхема 74182 позволяет получить до четырех наборов «входа переноса» и функций «формирования переноса» и «распространения переноса» и обеспечивает три сигнала выхода переноса, требуемые АЛУ, а также вспомогательными функциями следующего уровня.

Вспомогательные функции, вырабатываемые схемой ускоренного переноса, допускают последующие уровни ускоренного переноса. Вспомогательными логическими функциями в случае активного высокого уровня являются «отрицание формирования переноса» и «распространение переноса» Они обозначены через *X* и *У* соответственно и соединены таким же образом, как в случае с активным низким уровнем В данной логической конструкции вспомогательные функции используются для формирования трех сигналов «выход переноса» и двух вспомогательных функций, требуемых для добавочных уровней ускоренного переноса.

Микросхема 74181 не только обеспечивает 32 режима работы АЛУ в зависимости от состояния управляющих сигналов на входах M,S0...S3, но также допускает наращивание разрядности (вход CN и выход CN<4 для переносов).

Соответствующая схема сквозного переноса, имеющая четыре индицируемых разряда, с использованием микросхемы 74182 показана на рис.182. (Обычно, когда говорят о микросхемах, опускают две первые цифры, характеризующие серию или печатают апостроф. Таким образом, две рассмотренные выше микросхемы имеют номера 181 и 182. В связи с этим необходимо отметить, что совпадение номеров микросхем с номерами двух последних рисунков случайное Вероятность этого сложного события можно оценить по известной формуле Бернулли: она ничтожно мала, но факт, как говорится, на лицо и против него не «попрешь» – шутка!)

В связи со сложностью отображения на одном экране без прокрутки больших схем здесь режимы работы АЛУ специально подобраны так, чтобы происходило последовательное переполнение разрядов. В результате при суммировании единица как бы «пробегает» от первого АЛУ до четвертого (см. рис. 182). Задание операндов на первом АЛУ выполнено ключами, а на остальных с помощью заземления (лог. 0) и схемного компонента 1>—— (лог.· 1). Последний выбирается командами: Component>Didital Primitives> Stimulus Generators>Fixed Digital, а величина (Value) его напряжения выбирается равной 1 В.

Вообще же можно смоделировать работу АЛУ в динамике, задав входные слова на все разряды генераторами цифровых сигналов типа Stim или FStim (источник, записанный в файле), но это потребует их специального описания.

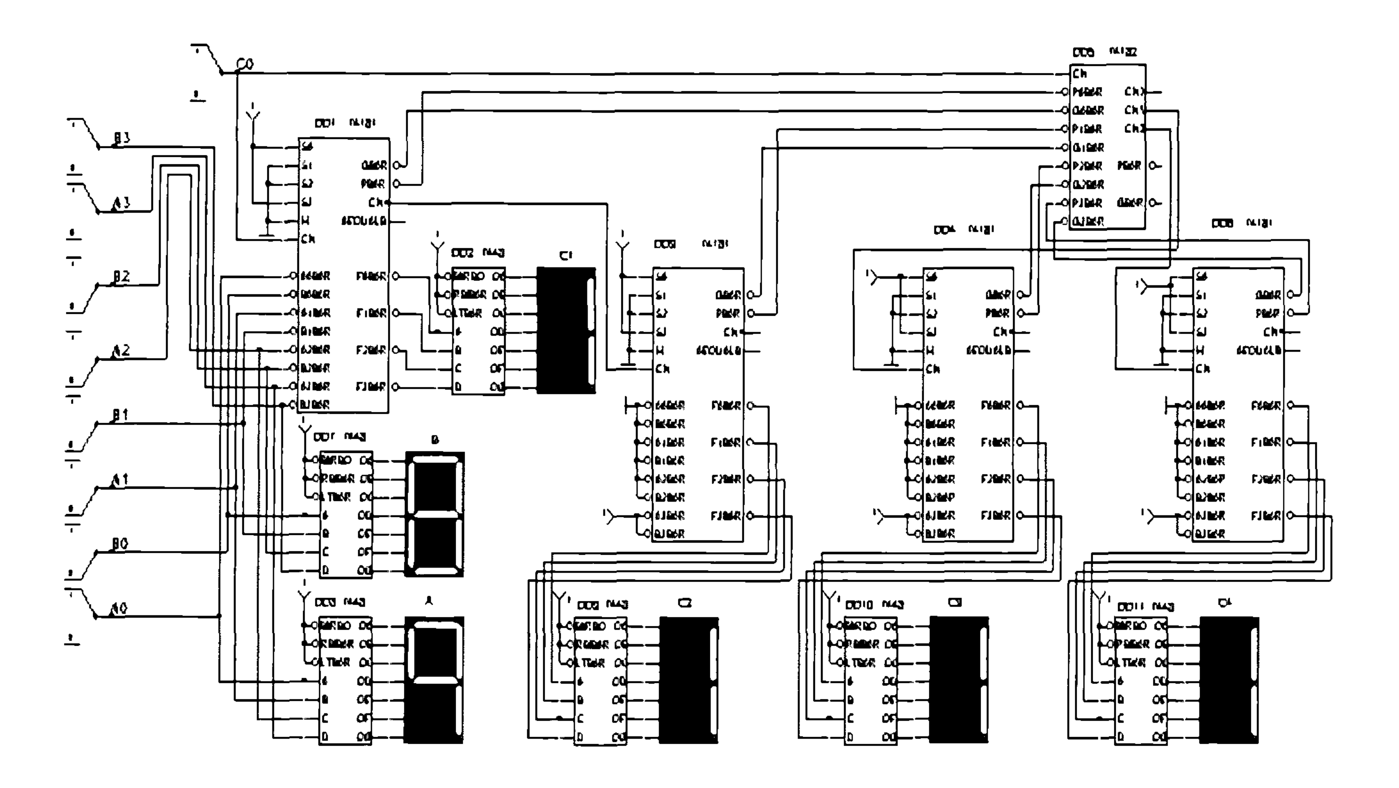


Рис. 182. Схема сквозного переноса АЛУ на микросхеме 74182

4.2. Запоминающие устройства

Память – это такое устройство в мозгу при помощи которого мы все забываем
А. Чейз

Цифровые запоминающие устройства (ЗУ), сокращенно называемые также памятью, служат для хранения информации и обмена ею с другими устройствами. Это, безусловно, одно из важнейших ЦУ, определяющих многие характеристики ПК. Для кратковременного хранения кодовых слов используют регистры, а для длительного хранения и для больших объемов информации служат специализированные микросхемы. Поскольку далее речь пойдет только о полупроводниковой памяти, то отметим, что микросхемы памяти занимают до 40% от общего выпуска микросхем.

Важнейшими характеристиками ЗУ являются их информационная емкость, организация и быстродействие.

Основным классификационным признаком ЗУ является способ доступа к данным (способ обращения к массиву элементов памяти), по которому они подразделяются на адресные, последовательные и ассоциативные.

Адресные ЗУ в свою очередь подразделяются на RAM (Random Access Memory – память с произвольным доступом) и ROM (Read-Only Memory – память только для чтения). В отечественной литературе эти типы ЗУ называются соответственно Оперативным Запоминающим Устройством (ОЗУ) и Постоянным Запоминающим Устройством (ПЗУ) Первые, как правило, энергозависимы, а вторые, напротив – энергонезависимы. Среди ОЗУ различают устройства статические – Static RAM (SRAM) и динамические – Dynamic RAM (DRAM).

Статические ПЗУ

В ЗУ с произвольным доступом для хранения каждого бита информации используется отдельный запоминающий элемент – ячейка памяти (ЯП). Обычно в этом качестве используются различного рода триггеры. Двоичная информация, занесенная в подобную ЯП, хранится там сколь угодно долго, пока не будет заменена другой или не будет отключено питание (энергозависимая память).

В зависимости от способа, которым находится каждая ЯП в массиве, различают структуры с одномерной (линейной) и двумерной адресацией.

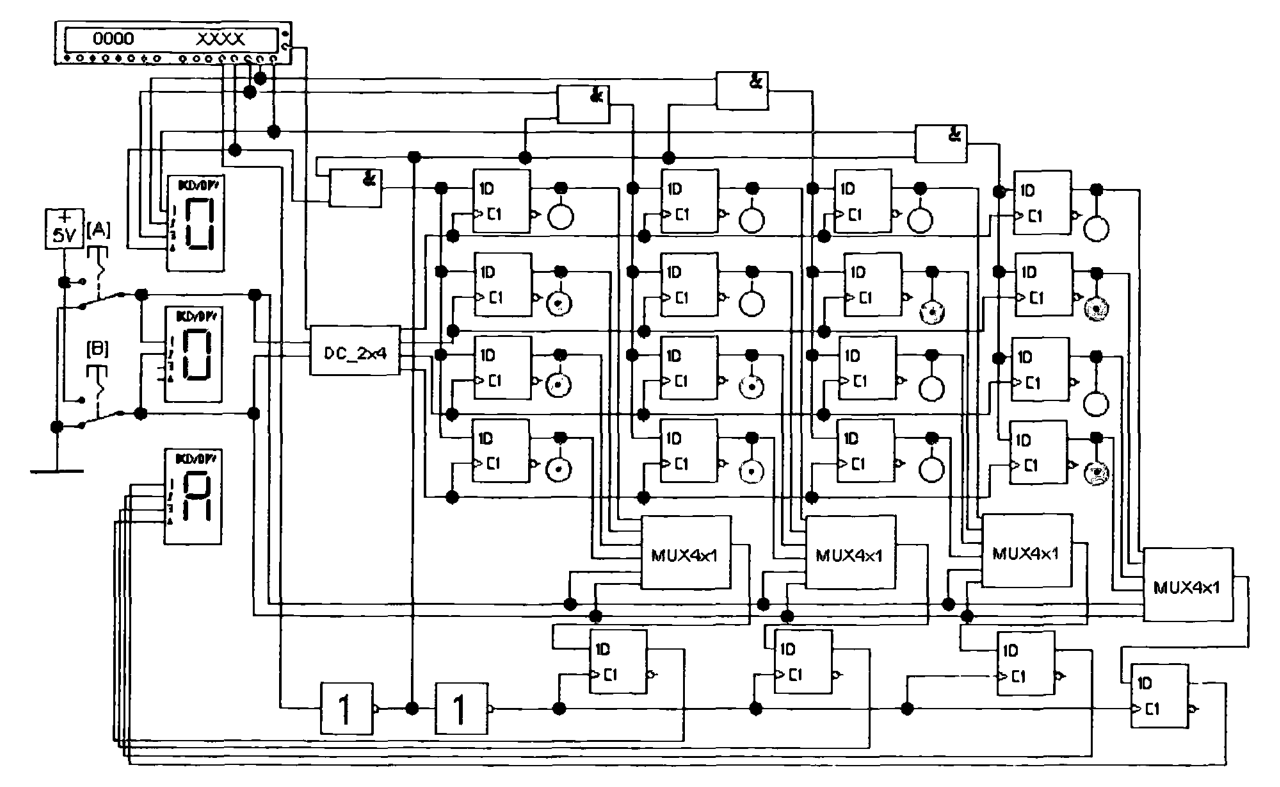


Рис. 183. Логическая структура ОЗУ с одномерной адресацией (**EWB**)

На рис. 183 в программе **EWB** представлена модель SRAM с линейной (пословной) адресацией. Здесь 16 ЯП из D-триггеров образуют квадратную матрицу 4x4. Через соответствующий вход D в ЯП вводится информация, по команде на входе C Информация хранится на выходе Q или считывается с него по аналогичной команде. Адресация к ЯП производится ключами A и B, и далее декодером DC_2×4. Заносимые в конкретные ячейки биты информации набираются на генераторе слов. Тактовый сигнал с этого же генератора используется для синхронизации триггеров, декодера и мультиплексоров, а выход 5-го разряда – для формирования сигнала типа разрешение записи/чтения. Записи соответствует лог. 0, а чтению – лог. 1. Вывод информации (чтение) производится по столбцам через мультиплексоры MUX4×1 Для наглядности работы модели она снабжена тремя дисплеями, которые (сверху вниз) отображают: вводимое слово, адрес ЯП и читаемое слово. Кроме того, в каждой ЯП поставлен индикатор ее состояния. Разумеется, это всего лишь иллюстративная, но работающая модель. На рис. 183 показано, что ключи A=B=0, т.е произведено считывание из верхнего ряда ЯП, занесенного туда ранее слова $(1100)_2$=A_{16}= 10_{10}. В данной системе считанная информация исчезает из памяти: как бы нельзя два раза войти в одну и ту же реку.

При необходимости побитовой записи/считывании информации используют структурную организацию памяти в виде двумерного массива-матрицы A×B с ЯП, которые можно выбирать на пересечении соответствующих строк и столбцов. Простейший D-триггер не позволяет сделать такого целенаправленного выбора. Логику работы ЯП надо видоизменить так, чтобы она позволяла проводить независимо эти операции

На рис. 184 показан вариант организации подобной ЯП, выполненный средствами программы **EWB.** Помимо логики выбора ячейки снабжены индивидуальной индикацией их состояний и выводимой информации. Адрес конкретной ячейки в массиве 4×4 задается ключами [0],[1] – для выбора номера строки (от A0 до A3) через дешифратор DCA и ключами [2] и [3] – для выбора номера столбца (от B0 до B3) через дешифратор DCB. Сигнал разрешения записи/чтения (W/R – Write/Read) подается ключом [Space]. На рис. 184,а показана организация соответствующих линий и ЯП с номером 0. Эту ячейку оформим как базовую ЯП и назовем по-английски BSC (Bit Storage Cell – двоичная запоминающая ячейка), представив ее в виде субблока (рис 184,б) Пользуясь этими заготовками, соберем SRAM с двумерной адре-

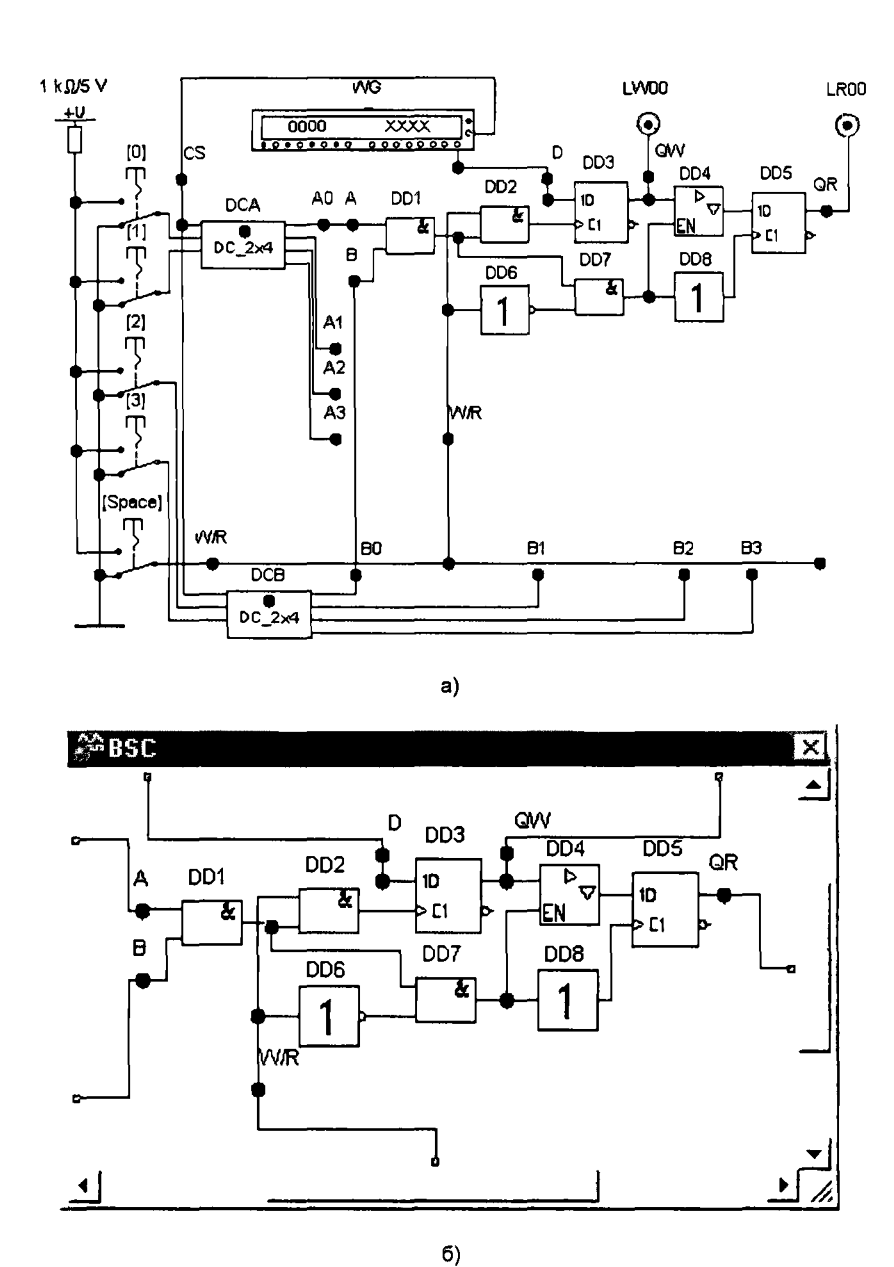


Рис 184. Ячейка памяти для памяти с двумерной адресацивй (**EWB**)

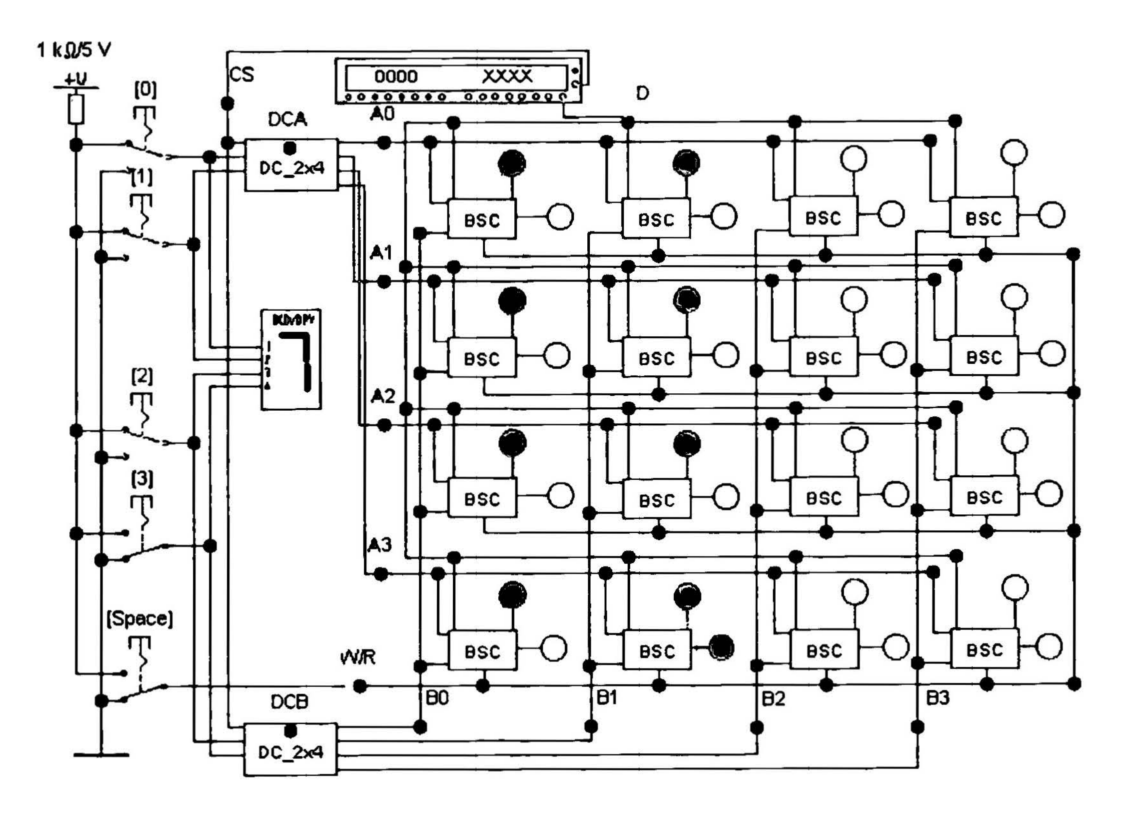
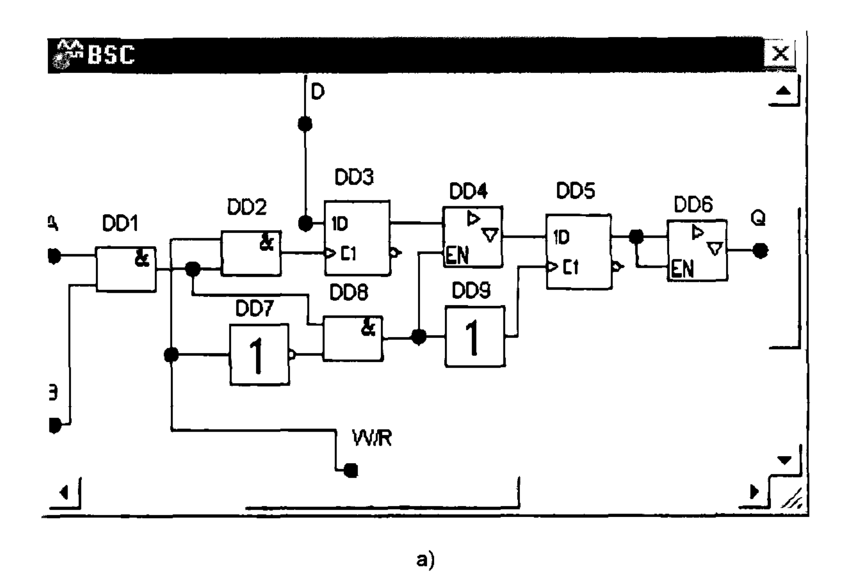


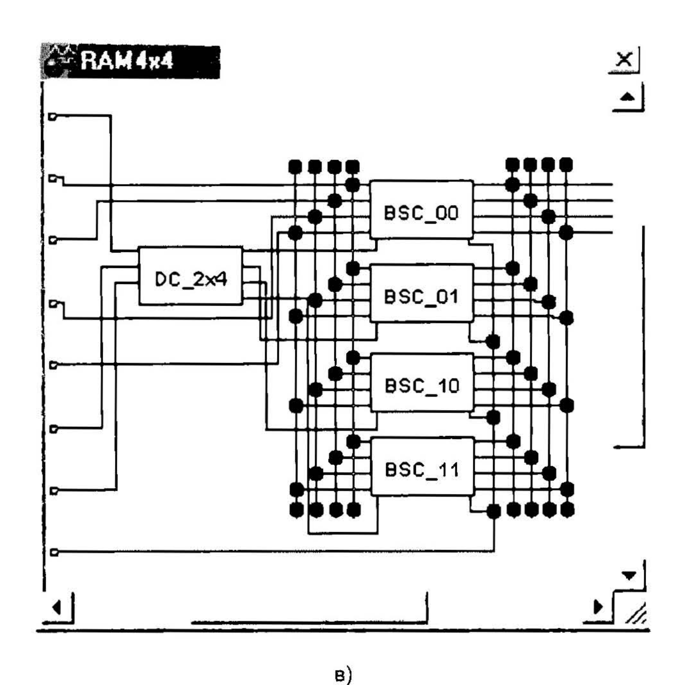
Рис 185   Логическая структура ОЗУ с двумерной адресацией (**EWB**)



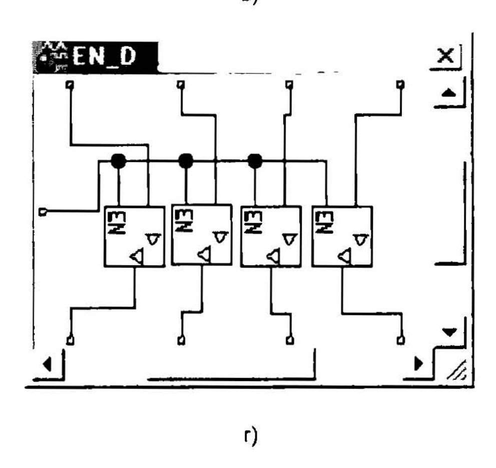
а)

б)

Рис. 186 (начало)

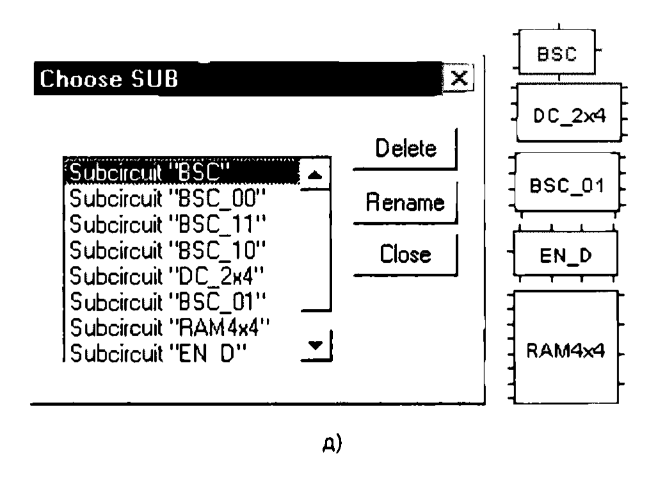


в)



г)

Рис. 186 (продолжение)



д)

Рис 186 (окончание) Субблоки компонентов ОЗУ для микропроцессора (**EWB**)

сацией (см рис 185). Здесь на дисплее показывается номер выбираемой ячейки. В данном случае это 7 (счет идет с нуля), т.е при выбранной системе нумерации это ЯП с адресом А3В1 Горящие на этой ячейке индикаторы показывают, что в ячейке хранится единица, и она была выведена для чтения.

Пользуясь описанными приемами и компонентами, создадим модель ОЗУ для дальнейшего использования в модели микропроцессора. Ограничимся памятью 4×4 с пословной организацией, но выполним ее так, чтобы информация не разрушалась после ее считывания и была обеспечена развязка с соответствующими линиями. Отдельная ячейка памяти в виде субблока BSC (см. рис. 186,а). Далее из четырех подобных ячеек создадим четыре «строчных» элементов с соответствующими поразрядными адресами: BSC_00, BSC_01, BSC_10 и BSC_11 (на рис. 186,б показан элемент BSC_00). Собрав из этих компонентов с помощью линий требуемую логическую структуру и дополнив ее дешифратором адреса, получим RAM4×4 (см. рис. 186,в). Для работы на общую шину данных создадим также субблок EN_D с четырьмя буферами шины, имеющими три состояния выхода (см рис. 186,г). Все созданные компоненты показаны на пользовательской панели выбора субблоков (Choose SUB) и в виде соответствующих схемных УГО (см. рис. 186,д). Работа модели ОЗУ из указанных компонентов показана на рис. 187.

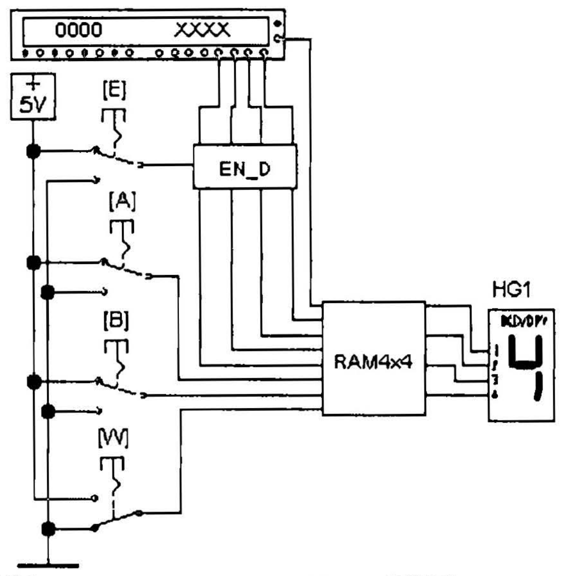


Рис. 187. ОЗУ для модели микропроцессора (**EWB**)

Рис 188 ОЗУ на микросхеме 74170 (**MC**)

В библиотеке цифровых компонентов программы **МС** имеется модель ОЗУ на микросхеме 74170. Данная микросхема содержит ОЗУ с произвольной выборкой объемом 16 бит (4 слова по 4 бита каждое: 4×4), которое позволяет одновременно записывать и считывать информацию. Работа этой микросхемы показана на рис. 188. Записываемое четырехразрядное слово подается на информационные входы D1–D4. Выбор ячеек производится подачей соответствующих уровней на адресные входы WA и WB, а чтение, соответственно, на RA и RB. Разрешение чтения и записи происходит при GWBAR=GRBAR=0. В рассматриваемой модели (см. рис. 188) слова генерируются в двоичном коде функциональным генератором G1 и показываются на индикаторе HL1, а выводятся из памяти с выходов Q1–Q4 на индикатор HL2. При считывании хранящаяся информация здесь не стирается. Однако наличие открытого коллектора и способ моделирования в данной программе приводят к некоторым особенностям поведения модели. На практике подобную микросхему используют в качестве быстродействующего (время выборки 20 нс) буферного запоминающего устройства.

4.3. Программируемые устройства

Мы только зайдем сюда, чтобы ты мог сказать, что побывал в лабиринте, но это совсем не сложно. Мы походим здесь минут десять, а потом отправимся завтракать.

Джером К. Джером
Трое в лодке, не считая собаки

Развитие цифровой техники исторически происходило так, что в системах обработки информации локализовывались отдельные функциональные части, которые интегрировали в себе большое число компонентов. Когда уровень интеграции достиг значений, позволяющих реализовать на одном или нескольких кристаллах требуемые программные алгоритмы, появились микропроцессоры и их упрощенные варианты для систем управления – микроконтроллеры. Эти устройства были стандартизованы и размещались на одной или нескольких больших или сверхбольших интегральных микросхемах.

Наряду с подобными стандартизованными и специализированными микросхемами в любой достаточно сложной системе с необходимостью присутствуют и нестандартные менее интегри-

рованные компоненты, такие как схемы управления, интерфейсные устройства и т.п. Реализация этой «нестандартной» части систем проводилась на микросхемах малого и среднего уровней интеграции. В результате с увеличением сложности систем, несмотря на большую интеграцию их основных частей, росло общее число корпусов микросхем, ухудшались быстродействие, надежность и другие характеристики. Наличие нестандартных компонентов увеличивало также и стоимость комплектов.

Решение возникшей дилеммы между специализацией и универсальностью привело к разработке больших и сверхбольших микросхем с программируемой и репрограммируемой структурой Необходимо подчеркнуть, что смысл термина «программирование» применительно к данным устройствам, в отличие от микропроцессоров и микроконтроллеров, где он фактически означает реализацию заданного алгоритма обработки входных кодов, имеет иное содержание. Здесь речь идет о программировании на аппаратном уровне, а именно, о возможности программного изменения внутренней структуры микросхем, так, чтобы оно обеспечивало реализацию наперед заданных логических функций. Структурно-программируемое устройство, таким образом, становится и универсальным и специализированным. Не следует, конечно, заблуждаться, что будто бы этот компромисс был достигнут бесплатно: «... сколько у одного тела отнимается, столько присовокупится к другому», – говорил мудрый Михайло Васильевич Ломоносов. Ценой, заплаченной за решение проблемы, является аппаратная избыточность на внутреннем уровне микросхем.

Известны различные типы структурно-программируемых устройств. Вначале надо назвать постоянные запоминающие устройства масочного типа (ПЗУ – ROM) и их производные: программируемые (ППЗУ – PROM) и репрограммируемые (РПЗУ). Хотя обычно их и не относят к данному классу устройств, однако, при ближайшем рассмотрении сразу видно: что и от чего произошло.

Первыми же в прямом смысле представителями данного класса устройств, благодаря которым их выделили в отдельное направление, явились программируемые логические матрицы (ПЛМ) – соответственно на английском языке – Programmable Logic Array (PLA). Кстати, в английском языке прилагательное programmable составное. program+able, что дословно означает – способный (able) к программированию. Здесь, в отличие от русского языка, исходное слово «программа» – program – пишется с одной m. В русском языке появляется лишняя м, а оттенок спо-

собности к возможности производить программирование этих устройств, при переводе почти исчезает.

Далее последовала программируемая матричная логика (ПМЛ) – соответственно на английском языке – Programmable Array Logic (PAL) и базовые матричные кристаллы. Разумеется, в данном контексте PAL – это отнюдь не Phase Alternation Line (строка с переменной фазой), которая является одним из общеизвестных стандартов цветного телевидения.

Устройства PLA и PAL в английской терминологии объединяются одним термином PLD (Programmable Logic Devices) – программируемые логические устройства. Поскольку эти устройства реализуются по интегральной технологии, то в отечественной литературе они именуются «Программируемыми Логическими Интегральными Схемами» – ПЛИС. По сути дела, сейчас сформировалось и активно развивается новое научно-техническое направление: проектирование и создание цифровых систем на основе ПЛИС, в рамках которого решаются задачи синтеза сложных комбинационных схем, конечных и микропрограммных автоматов.

Полупроводниковые ПЗУ

В масочных ЗУ типа ROM (M) информация записывается при изготовлении микросхем на промышленных предприятиях с помощью специального шаблона – маски, на финишной стадии процесса. В ЗУ типа PROM подобная запись осуществляется потребителем изделий (полуфабрикатов) с помощью специальных программаторов. Собственно программирование ПЗУ заключается в определенном размещении элементов электрической связи между горизонтальными и вертикальными линиями (проводниками) матрицы запоминающих элементов.

Элементами связи в масочных ЗУ служат диоды, биполярные и КМОП транзисторы и др.

Рассмотрим модель простейшего диодного ПЗУ, собранную в программе **EWB** (см. рис. 189). Матрица в левой части схемы и является подобным устройством. Топология («безразмерная» геометрия) ЗУ представляет собой 4 вертикальных проводника A, B, C, D и 16 горизонтальных (сверху вниз от 0 до 15 или в двоичной системе от 0000 до 1111), связанных в определенных местах диодами Диоды расположены не по случайному закону, а так, что если горизонтальные линии считать входами этого логического устройства, а вертикальные – выходами, то оно становится своеобразным преобразователем кода Вся остальная часть схе-

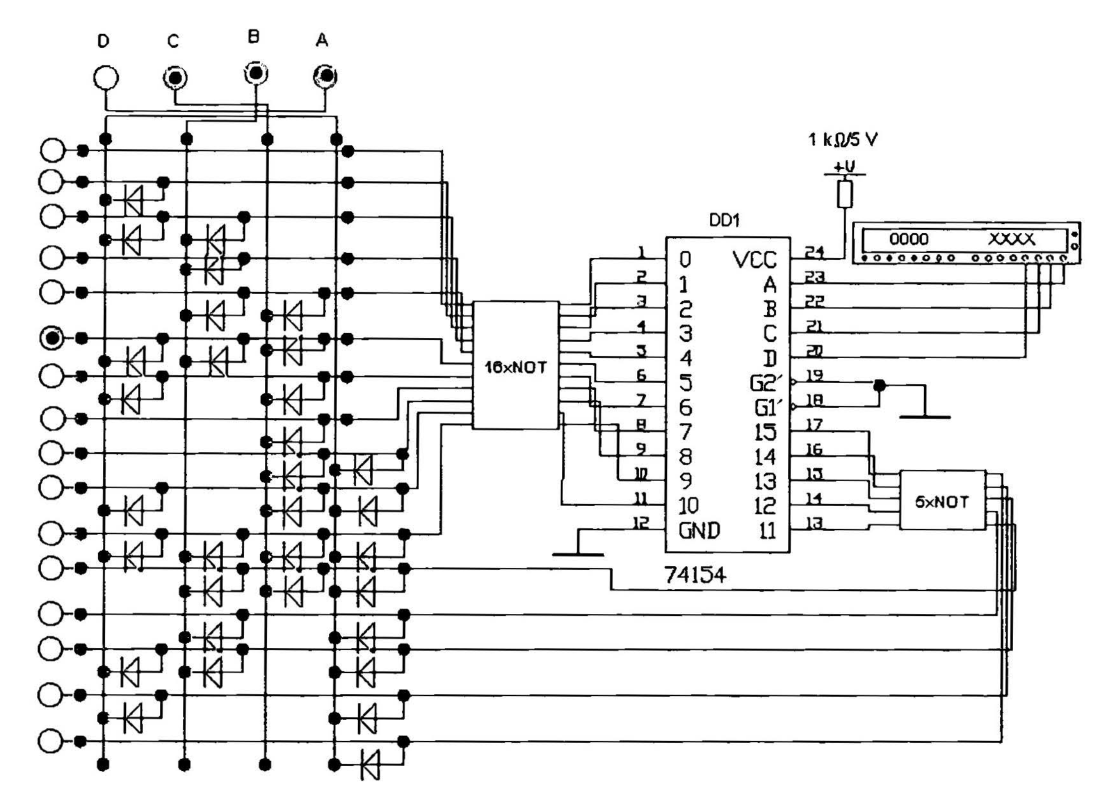


Рис. 189. ПЗУ на диодах (**EWB**)

мы к ПЗУ отношения не имеет, а служит для демонстрации его работы. Таким образом, с помощью диодной логики в ПЗУ «зашита» определенная таблица истинности.

Поскольку получаемый на выходе код представляет самостоятельный интерес, то дадим короткий комментарий по этому поводу. Как было отмечено ранее, в ЦУ встречается и недвоичное кодирование состояний (хотя и выражаемое двоичными числами), например в счетчиках Джонсона.

Представим себе некоторое фотоэлектрическое устройство, вырабатывающее кодовую последовательность при переходах типа освещен – затемнен. Соответственно, освещенному устройству соответствует кодовая 1, а затемненному – 0. Последовательность подобных переходов даст двоичные числа. Проблема возникает при переходе от одного числа к другому. Если счет вести в натуральном двоичном коде, то, например, при переходе от $0111_2=7_{10}$ к $1000_2=8_{10}$ должны поменяться цифры во всех разрядах, и вероятность ошибки здесь крайне велика. Теперь закодируем информацию так, чтобы последующие и предыдущие числа отличались только в одном самом старшем разряде. В рассматриваемом примере: $0100_G=7_{10}$ к $1100_G=8_{10}$. Тогда, при нахождении фотопреобразователя между 7 и 8 (т.е. между светом и темнотой), последняя цифра старшего разряда может быть определена с погрешностью не более половины интервала квантования. В итоге на выходе можно получить либо 0100_G, либо 1100_G. Эту ошибку в свою очередь можно сделать сколь угодно малой (в техническом смысле) за счет уменьшения шага квантования. Поскольку в ЦУ исключается необходимость одновременного переключения многих разрядов, то данный код является помехозащищенным.

Код, подчиняющийся указанной закономерности, известен в математике с 70-х годов XIX в. и относится к группе так называемых рефлексных кодов. Однако, активное использование этого кода началось только с 50-х годов XX в. после работ Ф. Грея (F. Gray), применившего его для построения преобразователя угловых перемещений в цифровой код. Этот код, называемый кодом Грея (Gray code), широко используется в аналого-цифровых преобразователях. Отсюда – нижний подстрочный индекс G в обозначениях приведенных выше чисел)

Таблица соответствия десятичного и двоичного кодов с кодом Грея такова.

Таблица 4

Код								
Десятичный	Двоичный				Грея			
№	d	c	b	a	D	C	B	A
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

Анализируя приведенную таблицу, можно обнаружить различные симметрии, свойственные коду Грея. Отметим только одну из них: если склеить нижний ряд чисел с верхним в кольцо (цилиндр), то закономерность, связанная с изменением старшего разряда, останется в силе, т.е. это полный циклический код.

Вернемся теперь к рассмотрению работы ПЗУ, показанного на рис.189. Необходимые числа задаются в четырехразрядном двоичном коде с выхода генератора слов на четырехразрядный дешифратор/демультиплексор 4×16. На одном из 16 выходов микросхемы, соответствующем десятичному представлению этого числа, возникает сигнал низкого уровня. К сожалению, в УГО 74174 программы **EWB** выходы показаны прямыми. Для инверсии выходных сигналов используем 16 инверторов (NOT), образовав из них подходящие субблоки. С выходов инверторов сигнал подается на одноименные по номерам горизонтальные проводники ОЗУ и одновременно для наглядности на левый ряд СИД и вы-

ходными линиями приводит к тому, что сигнал, проходя по некоторым (прямовключенным) диодам, поступает на определенные выходные линии. В подобном случае загораются соответствующие СИД верхнего ряда. Так, на рис. 189 показан случай, когда число на генераторе соответствует 0101_2, горит в вертикальном ряду СИД №5 (самый верхний СИД принят за нулевой), т.е. декодирование в десятеричную систему дает число 5_{10}, и в верхнем (выход ПЗУ) читаем в коде Грея число 0111_G.

Полученный результат полностью соответствует приведенной выше таблице Как уже отмечалось, выполненное в виде микросхем подобное устройство может использоваться, например в аналогово-цифровых преобразователях Но мы пока не ответили на вопрос, а в чем же собственно заключалось программирование ОЗУ?

Для этого надо рассмотреть обратную задачу. Так вот пусть необходимо создать подобное ОЗУ. В соответствии с ТИ можно составить схему межсоединений в матрице – схему ее «прошивки». Последний термин достался нам в наследство от ПЗУ на ферритовых кольцах, в которых процесс программирования напоминал вышивание, поскольку провода пропускались через определенные кольца, да и выполняли эту работу женщины.

Для того чтобы как в рассмотренном примере при входе на горизонтальную линию №5 сигнал выходил на вертикальных линиях A, B, C, необходимо, чтобы между этими линиями были прямо включены соответственно три диода Это соответствует диодной логике *ИЛИ*. В то же время сигнал в этом случае не должен попадать на линию D

При масочном способе диоды выращивают на нужных местах по специальной маске При программировании ОЗУ в изначально полностью заполненной матрице (когда все входные линии связаны диодами со всеми выходными) ряд связей надо ликвидировать, только и всего. Таким образом, как бы вся память была забита единицами и некоторые из них надо стереть. Однако не следует особенно обольщаться кажущейся простотой. Скорее следует вспомнить, что и самая распрекрасная скульптура, по выражению Микеланджело, – это всего лишь глыба мрамора, из которой скульптор убрал все лишнее. Если же Вам покажется, что сравнение с искусством здесь неуместно, то замечу лишь, что одна из лучших практических книг по электронике, которую мы неоднократно цитировали, называется «Искусство схемотехники» (а в оригинале еще более изящно: «The Art of Electronics»).

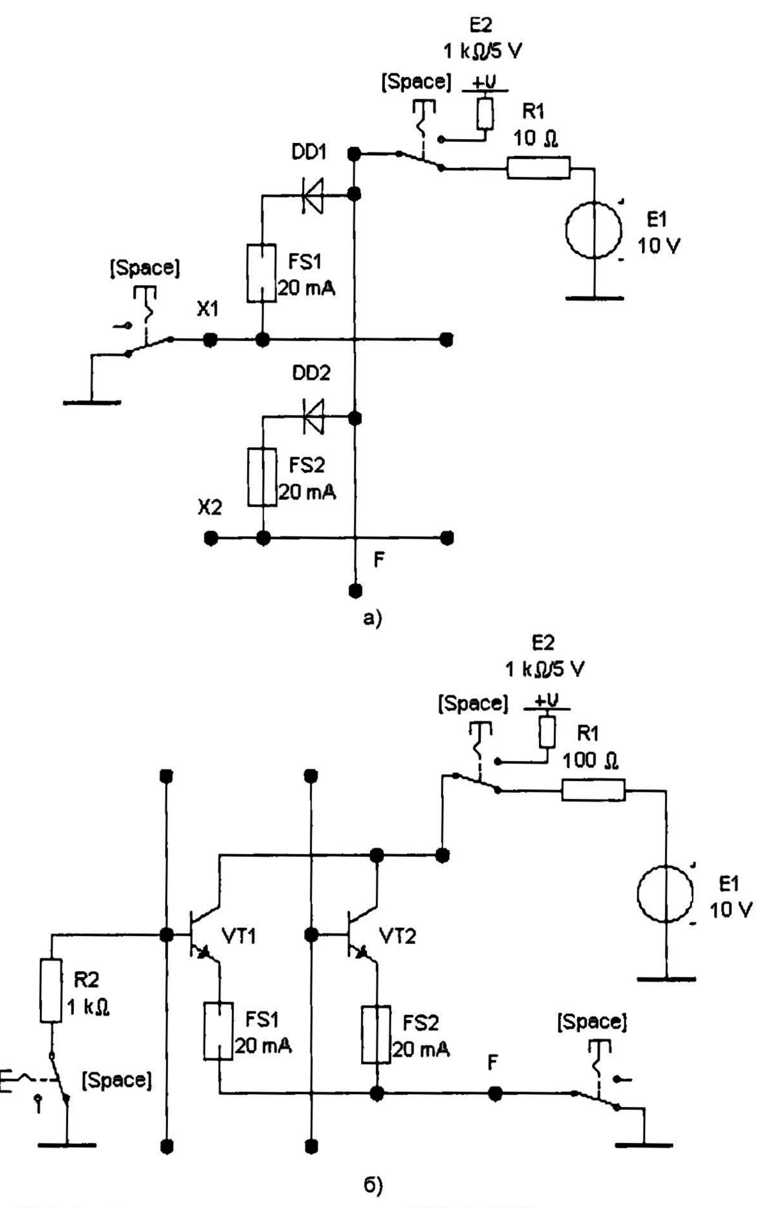


Рис. 190. Фрагменты программируемых ПЗУ (**EWB**)

Смоделируем процесс программирования, выделив отдельный фрагмент матричной структуры *ИЛИ* (см. рис. 190,а) из двух диодов DD1 и DD2, последовательно с которыми включены плавкие вставки FS1 и FS2 (от FUSE – плавкий предохранитель). Пусть номинальный ток вставок на порядок меньше максимально допустимого для диодов (т.е. они должны «перегореть» раньше, чем диод выйдет из строя при прямом включении).

Входные горизонтальные провода матрицы в этом небольшом фрагменте представлены входами X1 и X2, а выходные вертикальные – проводом F. Источник E2 представляет собой штатное питание микросхем, а источник E1 с повышенным напряжением и ключами [Space] здесь моделирует программатор. Предположим, что в этом фрагменте именно связь, образуемая диодом DD1, является лишней. Переключив ключи [Space] в нижнее положение, подаем на эту связь повышенное напряжение и пережигаем перемычку. Действуя аналогично по заранее разработанной программе, получаем необходимое ПЗУ.

Возвращаясь к понятию избыточности, видим, что а данном случае из 64 диодов нам пришлось бы устранить ровно полоаину

Обычно в качестве диодов в матрице используют биполярные (см. рис. 190,б) и МОП-транзисторы в диодном включении. Обычно это в масочные или программируемые ПЗУ «прошивают» алфавиты и стандартные функции. Перемычки изготавливают из высокоомных сплавов или поликристаллов. Используются также встречно-включенные последовательно соединенные диоды и другие элементы. Более совершенные технологии предусматривают прошивки ультрафиолетовым светом или электрическими полями.

Программируемая логика

Основой программируемых логических матриц и программируемой матричной логики (ПЛМ и ПМЛ) служит последовательность двух матриц: из элементов *И* на входе и *ИЛИ* на выходе, а также соответствующие буферные каскады.

Таким образом, в первой матрице осуществляются необходимые операции *И* над входными переменными в прямой и инверсной форме, а затем эти результаты в нужном сочетании выводятся второй матрицей за счет операции *ИЛИ*. Отсюда следует, что ПЛМ реализуют дизъюнктивную нормальную форму в двухуровневой логике. Это соответствует обычному выводу логических функций «по единицам» в таблице истинности Комбинации перечисленных операций и определяются программированием

Входные буферы преобразуют однофазный сигнал в парафазный, т.е. дают на горизонтальные линии матрицы прямой и инверсный сигналы. Эти сигналы поступают на диодные ячейки, собранные по схеме *И* (в отличие от рассмотренной выше диодной матрицы ПЗУ), в которых при прошивке оставлены необходимые связи, а затем собираются в матрице *ИЛИ*.

В ПМЛ матрица *ИЛИ* имеет жесткую логику, а программируется только матрица *И*. Это, безусловно, сужает возможности устройств. В ПЛМ – программируются обе матрицы, что дает возможность соединить любой конъюнктор с любым дизъюнктором. Программирование матрицы *ИЛИ* может быть выполнено, например, аналогично показанному на рис. 190,б.

Существенное расширение схемотехнических вариантов и соответственно возможностей программируемой логики достигается введением внешних соединений между матрицами *И* и *ИЛИ* и организацией обратных связей.

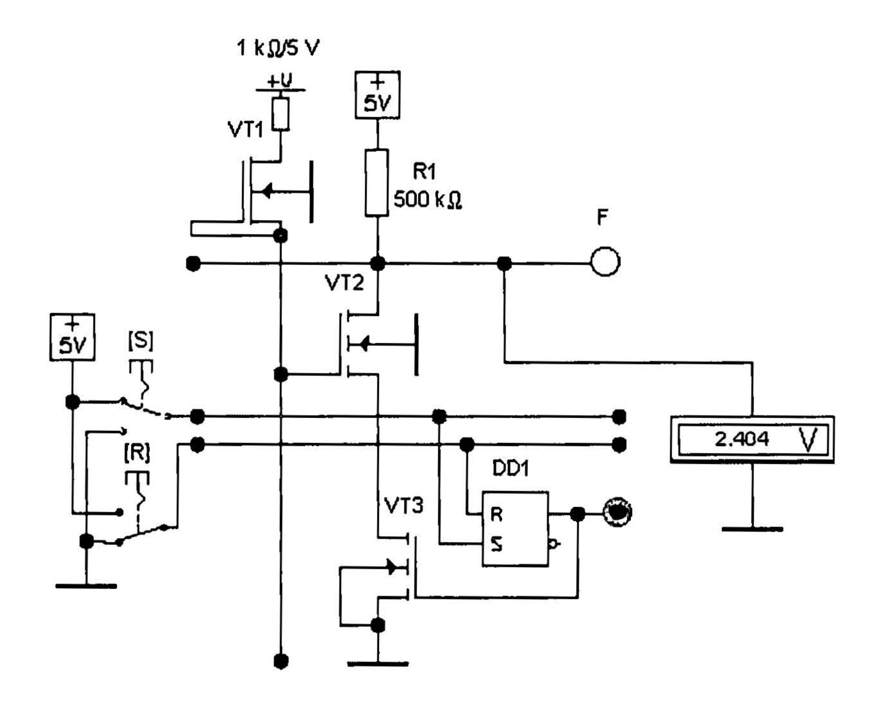


Рис. 191. Модель фрагмента ПЛИС с управляемым выходом (**EWB**)

В интегральной технологии ПЛИС, кроме того, в отдельные ячейки вводят дополнительные элементы, например триггеры Модель фрагмента ПЛИС, в ячейку которой введен триггер, показана на рис. 191. Здесь вольтметр поставлен для отладки режима. (Вообще же эта модель весьма капризна и, скажем, собирать на ее основе какие-либо устройства не очень удобно.) Управление триггером по линиям R,S разрешает или запрещает связь матричных ячеек *И* и *ИЛИ* Таким образом, ПЛИС с подобными ячейками позволяет гибко изменять (перепрограммировать) структуру устройства в процессе работы.

В программе **МС** имеются простейшие модели ПЛМ и ПМЛ устройств, на которых можно познакомиться с некоторыми особенностями их свойств. Зададим команды Components>Digital Primitives>Programmable Logic Array, после чего откроется выбор. PLAND33 – программируемые логические матрицы типа AND с тремя входами и тремя выходами (размером 3×3), PLOR33 – программируемые логические матрицы типа OR размером 3×3 и PLANDC33 – программируемые логические матрицы типа AND размером 3×3, но содержащие также внутри наряду со столбцами прямого кода и столбцы обратного кода.

Начнем с матрицы PLAND33, которая далее для сокращения записей обозначается как PLA и дополняется позиционным номером, например, PLA1 (см. рис. 192,а). Подсоединим к входам In0–In2 цифровые ключи, а к выходам – индикаторы. Структура матрицы в данном случае аналогична структуре ТИ. Здесь только надо отметить, что в данной программе нумерация входов инверсна по отношению к принятой нами ранее в ТИ поразрядной числовой нумерации.

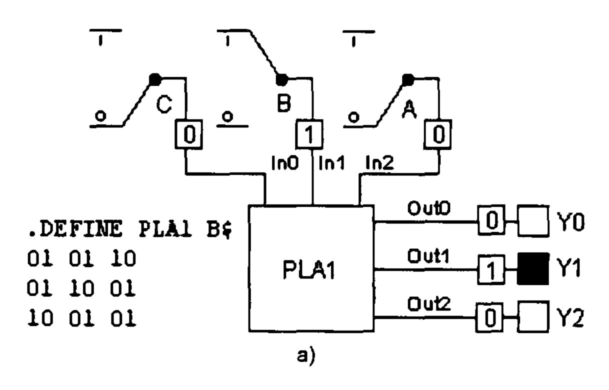
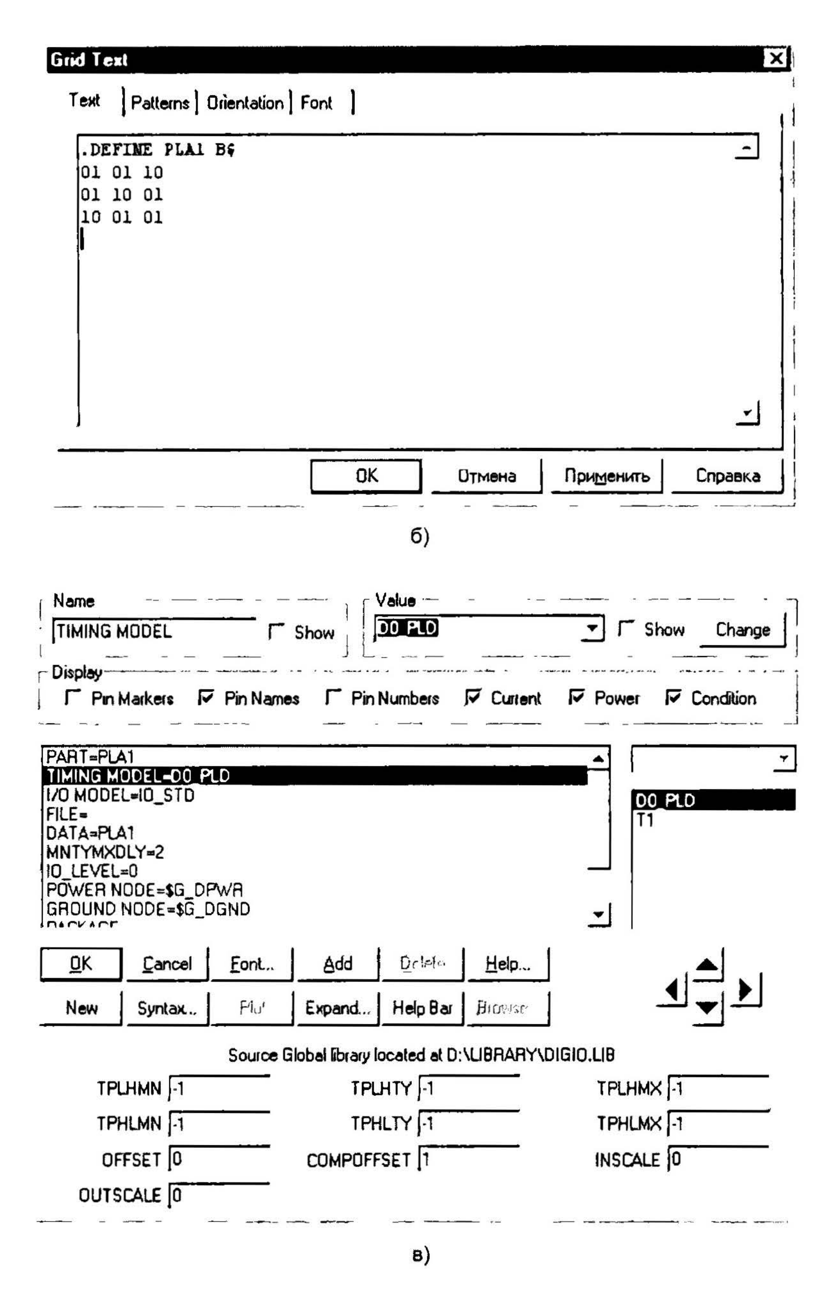


Рис. 192 (начало)



б)

в)

Рис 192 (окончание) ПЛМ типа AND (**MC**)

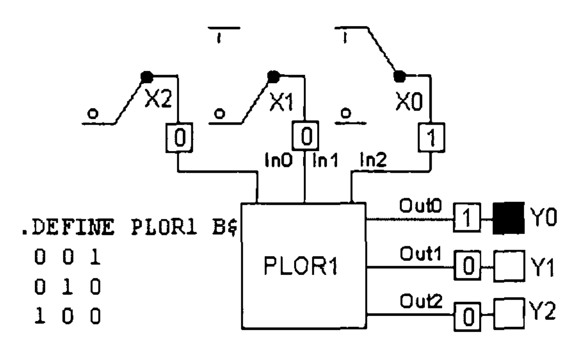


Рис. 193. ПЛМ типа OR (**МС**)

Для программирования матрицы надо как бы заполнить соответствующую ТИ вида:

Состояния соединений			Выходы	Состояния выходов
In2	In1	In0		
			Out0	Y0= In2& In1& In0
			Out1	Y1= In2& In1& In0
			Out2	Y2= In2& In1& In0

Изначально все выходы имеют значения лог. 1, и если ТИ будет пустой, то на всех выходах будут единицы. Как правило – это верный признак того, что вводимые данные не попадают в расчет. Язык, применяемый для описания состояний соединений по отношению к входным сигналам, имеет некоторые особенности. Лог. 1, т.е. наличие соединения, кодируется как 1 или 10, отсутствие соединения кодируется через 0 или 00, и, наконец, инверсия по отношению к входному сигналу как 01. В последнем случае 0 перед 1 заменяет штрихи в верхних индексах или черту сверху, обычно применяемые для обозначения инверсии. Поскольку выходной сигнал формируется операцией *И,* то при заполнении таблицы надо так задать состояния, чтобы, если требуется, получить единицу для определенного состояния входов, то и соответствующая операция по строке должна дать единицу. Например, если требуется, чтобы Y1=1 только при единице на выходе Y1=A`BC`=1 Саму программу кодировки матрицы можно

напечатать в текстовом окне, открыв его командой Ctrl+G или щелкнув ЛКМ по соответствующей надписи в левом нижнем углу основного окна [▶|\Text/Page 1/]. Возврат на рабочую страницу осуществляется повторной командой Ctrl+G или щелчком ЛКМ по соседней надписи Page Для отладки схем матрицу удобно держать перед глазами вместе со схемой, что можно выполнить, напечатав ее в режиме нанесения надписей на схему. Этот режим выбирается кнопкой **T** или командой Ctrl+T В появившемся окне (см рис. 192,б) печатается программа в формате

DEFINE [DATA=<флаг системы счисления>$<данные программы>$]

В нашем случае как флаг системы счисления используется латинская буква B (от Binary – двоичная), знаком доллара $ выделяется рабочая часть программы. (Американцы так любят этот знак, что ставят его где надо и не надо; с первым долларом придется смириться, а без второго программа работает, так что сэкономим и печатать его не будем) Очень важно, чтобы в окне свойств компонента (см. рис. 192,в) метка данных совпадала с принятой в текстовом окне. В рассматриваемом случае – это PLA1. Надо иметь в виду, что не допускается наличия в одном файле копий программ кодировки одного и того же устройства В подобном случае выдается сообщение об ошибке (Error) типа наличия дубликата (Duplicate. Define...).

Второй программируемой матрицей является *ИЛИ*, которую мы обозначим как PLOR1 (см. рис. 193). В этой матрице исходно все соединения отсутствуют, и при отсутствии программирования на выходе будут низкие уровни. Поскольку при операции *ИЛИ* лог. 1 будет на выходе, если хотя бы один вход будет соединен с выходом, то необходимо в соответствующих узлах образовать подобные соединения. Например, если по выходу Out1 надо иметь элемент 3*ИЛИ*, то в верхней строке нужно проставить три единицы (см. рис. 193).

Рассмотренные матрицы допускают в принципе расширение, как по входам, так и по выходам. На основе этих матриц можно построить различные комбинационные устройства.

На рис. 194 показан декодер 3×8 на ПЛМ типа PLAND33. Здесь три матрицы PLA1–PLA3 имеют индивидуальные программы соединений и включены параллельно по входу. Работа этого дешифратора полностью аналогична ранее рассмотренному на рис. 194.

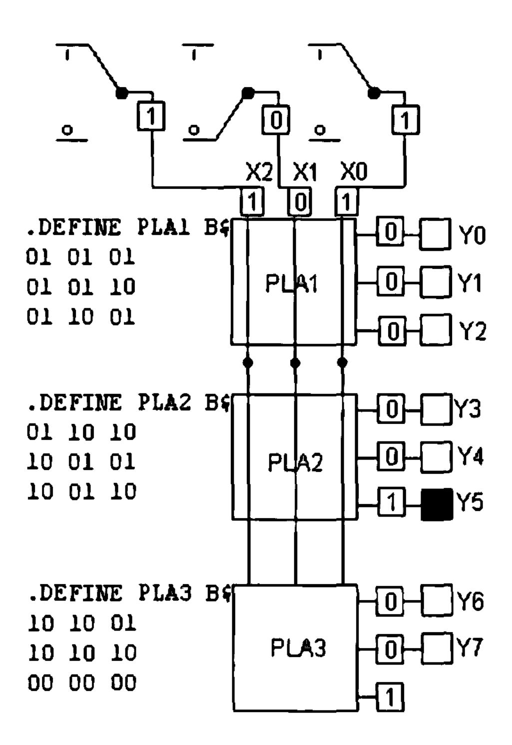


Рис 194 Дешифратор 3×8 на ПЛМ (МС)

На рис 195 показана структура мультиплексора 4×1, выполненного на матрицах типа PLAND33 и PLOR33. Этот мультиплексор также аналогичен ранее рассмотренному на рис. 104,а. Здесь на матрицах *ИЛИ* выполнено расширение по выходу. Фактически приведенная структура показывает модель типа ПЛМ: матрицы PLA1–PLA4 соответствуют программируемой матрице *И*, а матрицы PLOR1 и 2 – программируемой матрице *ИЛИ*.

Из приведенных схем может сложиться впечатление, что матричные устройства сложнее аналогичных, выполненных на отдельных интегральных компонентах. На самом деле это не так Во-первых, надо заметить, что программную модель 3×3 вряд ли можно назвать удачной по количеству входов и выходов. Во-вторых, с ростом сложности устройства они будут все больше превосходить свои аналоги по многим характеристикам

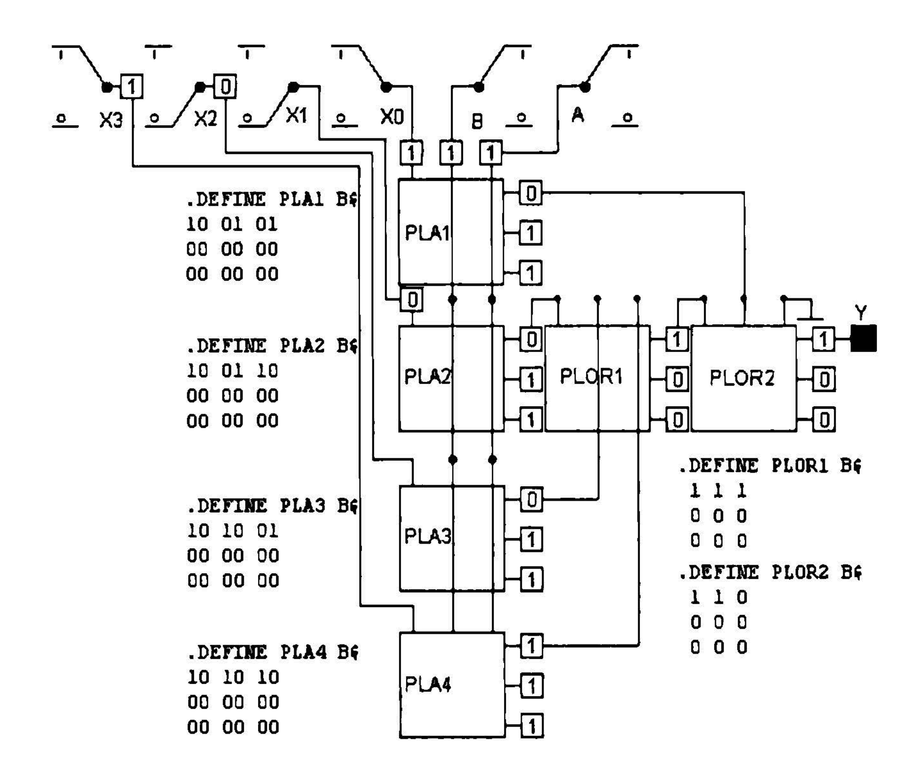
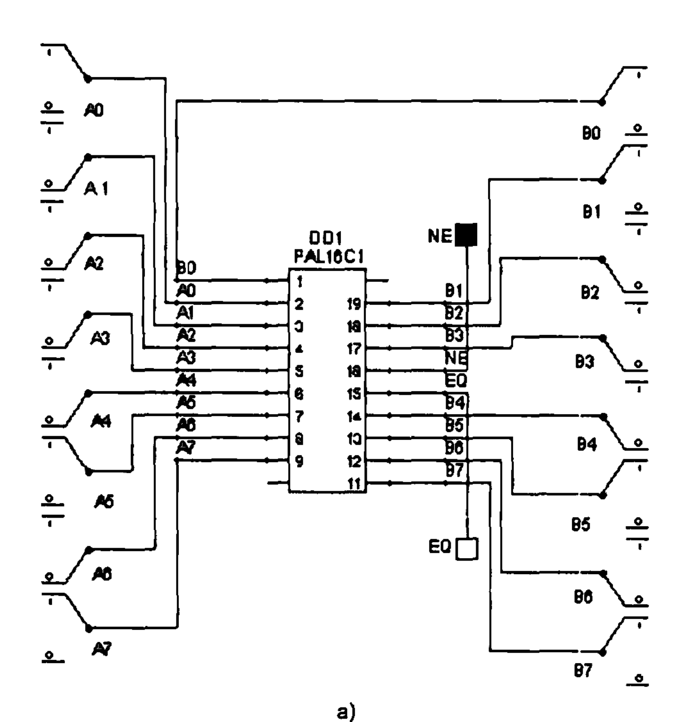


Рис. 195. Мультиплексор 4×1 на ПЛМ (**МС**)

В заключение приведем модель восьмиразрядного компаратора на ПЛИС типа PAL16C1 (см. рис.196,а). Этот компаратор сравнивает два восьмиразрядных слова. Если они равны, то 1 выводится на выход EQ, а если не равны, то – на NE. Программирование компаратора осуществляется в специальном текстовом файле JEDFILE, имеющем расширение .JED: «PAL16C1.JED» (см. рис. 196,б).

ПЛИС в настоящее время широко используются в качестве интерфейсных схем, а также в микропроцессорных системах при организации обмена информацией и стыковки различных микросхем между собой и устройствами ввода-вывода.



а)

8-Bit Equality Comparator*
QP20* QF512* F0*
L000
01101111111111111111111111111111
10011111111111111111111111111111
11110110111111111111111111111111
11111001111111111111111111111111
11111111011011111111111111111111
11111111100111111111111111111111
11111111111101101111111111111111
11111111111110011111111111111111
11111111111111110110111111111111
11111111111111111001111111111111
11111111111111111111011011111111
11111111111111111111100111111111
11111111111111111111111101101111
11111111111111111111111110011111
11111111111111111111111111110110
11111111111111111111111111111001*
C3BC4* 85E5

б)

Рис. 196. Восьмиразрядный компаратор на ПЛИС PAL16C1 (МС)

4.4. Микропроцессоры

«А не пора ли нам замахнуться
на Вильяма нашего Шекспира?!»
К/ф «Берегись автомобиля».

Да, наступило время «собирать камни». Весьма остроумно «камнем» радиолюбители называют микропроцессор, поскольку он выполнен из кристалла кремния...

За истекшие три десятилетия микропроцессоры прошли гигантский путь от первого чипа Intel 4004, содержащего 2300 транзисторов, работавшего на частоте 750 кГц и выполнявшего 60 тысяч операций в секунду, до современных микропроцессоров Intel, AMD, DEC и других ведущих компаний, содержащих десятки миллионов транзисторов, работающих на гигагерцевых частотах и выполняющих миллиарды операций в секунду.

Простейшая ЭВМ, рассматриваемая на аппаратном уровне, состоит из пяти функциональных блоков устройства ввода, микропроцессора (включающего АЛУ и устройство управления), памяти и устройства вывода Само собой разумеется, что присутствует и блок необходимого питания Связь между отдельными блоками осуществляется с помощью соответствующих шин

Под термином «шина» будем понимать совокупность линий (проводников), по которым передается информация или энергия от любого из нескольких источников к любому из нескольких приемников Поскольку вокруг этого термина в литературе не один десяток лет продолжается мягкая дискуссия, то, не вступая в нее, ограничимся лишь некоторыми историко-лингвистическими замечаниями. В русском языке само слово «шина» было употреблено еще М.В. Ломоносовым предположительно к «железному обручу на ободе колеса». Это было прямое заимствование из немецкого – Schiene, имеющего значения: шина, рельс, лубок; в английском языке подобное заимствование привело к слову Sheen – голень, приближающему нас к не очень приятной медицине. В отечественной литературе по вычислительной технике это немецко-русское слово было использовано как перевод английского –Bus Последнее является усеченным от «omnibus», представляющим в латинском языке слово omnis – «весь, всякий», имеющем во множественном числе дательного падежа окончание ibus. Так римские колесницы и повозки императора Клавдия, завоевавшего южную часть туманного Альбиона, превратились в английские омнибусы – общественный транспорт. Не зная латыни, абориге-

ны, отбросив основу слова, стали называть их лаконично – Bus, и позже перенесли это слово на название автобуса.

Многозначность (полисемия) языков мира и использование метафор для образования терминов в новых областях науки и техники никогда не оставят без труда лингвистов, переводчиков, редакторов и различные комитеты и комиссии по выработке стандартов. Например, в английском языке можно указать еще на ряд слов, имеющих в русском языке достаточно близкие (в функциональном отношении) к слову «шина» понятия: channel, magistral, path, route, trace, track, tract, trunk. Так что все дело в договоренности и используемом контексте.

В силу ряда особенностей моделирующих программ ограничимся здесь лишь сборкой гипотетической модели устройства для демонстрации принципа действия простейшей ЭВМ при выполнении некоторых функций. Построенная модель представлена на рис. 197.

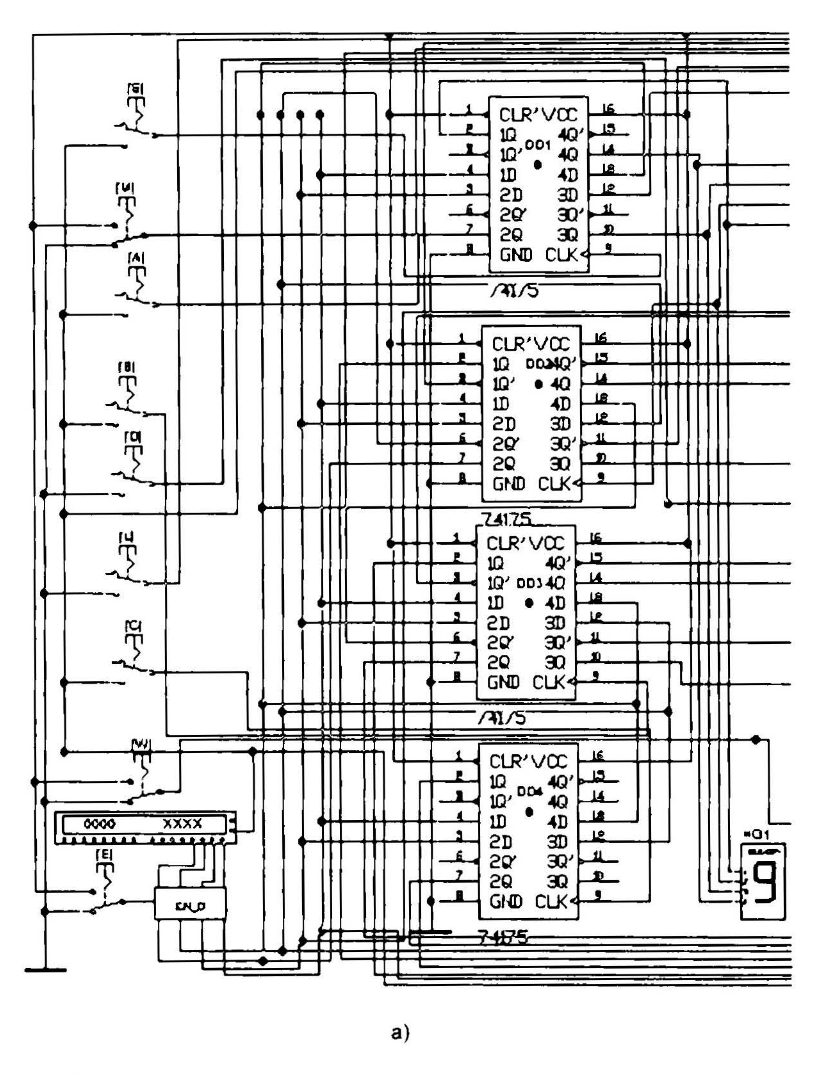
В левой части схемы (см. рис. 197,а) располагаются устройства ввода и управления в виде ключей и генератора слов. Далее следуют четырехразрядные регистры DD1–DD4 на интегральных схемах 74175. В правой части схемы (см. рис. 197,б) располагаются: АЛУ – DD5 на интегральной схеме 74181, инверторы – DD6 на интегральной схеме 7404, синхронный реверсивный программируемый десятичный счетчик – DD7 на интегральной схеме 74192, четыре двухканальных мультиплексора – DD8 на интегральной схеме 74257 и ОЗУ в виде рассмотренной выше RAM4×4. Вывод промежуточной информации и результата производится на индикаторы HG1–HG7. Четырехразрядная шина, представленная на рис. 197 в разнесенном виде, связывает генератор слов, регистры, мультиплексор, ОЗУ и выходной индикатор HG5 Ключами задают необходимые режимы работы отдельных блоков регистров, АЛУ, ОЗУ. Генератор слов выполняет одновременно роль ПЗУ, тактового генератора и программного счетчика

Приведем простой пример работы этой виртуальной модели ЭВМ Пусть требуется сложить два числа 2 и 3 Выполним следующие шаги (команды).

1 Выставим все ключи в верхнее положение

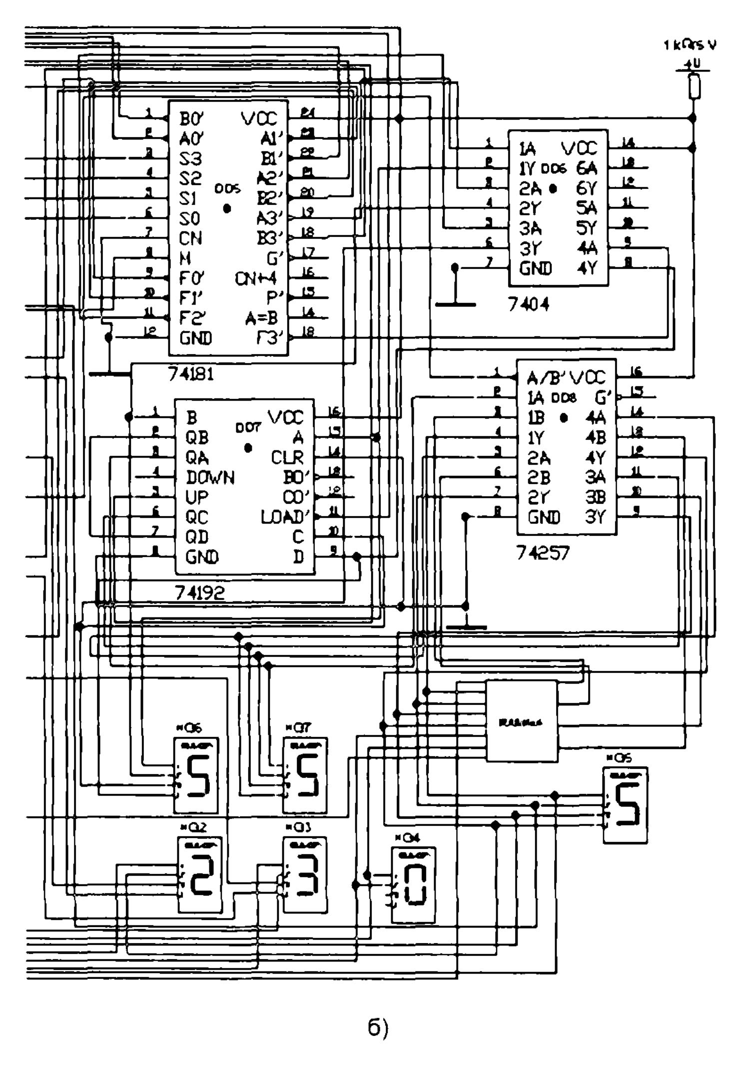
2 Откроем генератор слов, установим на нем пошаговый режим работы и зададим первые слова: 0000, 0009, 0002, 0003, 0000, 0000, 0000,

3 Переведем ключ S в нижнее положение, включим клавишу моделирования и нажмем на генераторе слов кнопку Step: на шине данных (дисплей HG5) появится цифра 9. Еще раз нажмем



а)

Рис. 197 (начало)



б)

Рис. 197 (окончание). Модель простейшей ЭВМ (**EWB**)

Step: на дисплее HG1 появится цифра 9, на дисплее HG5 – цифра 2.

4 Переведем ключ S в верхнее положение, ключ A – в нижнее положение и нажмем кнопку Step: на дисплеях HG2, HG6 и HG7 появится цифра 2, на дисплее HG5 – цифра 3.

5. Переведем ключ A в верхнее положение, ключ B – в нижнее положение и нажмем кнопку Step: на дисплее HG3 появится цифра 3, на дисплее HG5 – цифра 1.

6 Переведем ключ B в верхнее положение, ключ M – в нижнее положение и нажмем кнопку Step: на дисплеях HG6 и HG7 появится цифра 5

7 Переведем ключ C в нижнее положение, нажмем кнопку Step на дисплее HG4 появится цифра 0.

8. Переведем ключ C в верхнее положение, ключи E и W – в нижнее положение и нажмем кнопку Step: на дисплее HG5 появится цифра 5.

Прокомментируем эти шаги. На шаге 2 программируются значения, выводимые на шину данных на последующих шагах На шаге 3 производится начальное обнуление регистров и устанавливается режим работы АЛУ: цифре $9_{10} = 9_{16} = 1001_2$ соответствует режим арифметического сложения операндов при M=0. На шагах 4 и 5 в регистры заносятся операнды: A=2 и B=3. На шаге 6 в АЛУ производится операция сложения. На шаге 7 выбираются ячейки ОЗУ с адресом 00 и в них заносится результат сложения Наконец, на шаге 8 производится чтение из ОЗУ результата и его вывод на дисплей.

Совет **(EWB)** Не выключайте режим моделирования до окончания полного цикла работы программы. В случае ошибок в промежуточных шагах вернитесь к началу и, выключив режим моделирования, заново переустановите ключи.

На приведенной модели можно изучить и другие режимы работы ЭВМ Можно также трансформировать эту модель для изучения и более сложных программ.

* * *

Вот и пройден путь компьютерного моделирования от простейших логических элементов до ЭВМ Это, конечно, может послужить фундаментом для дальнейшей работы, ведь несмотря на все возрастающую сложность компьютеров основные их принципы остаются пока неизменными. Однако никогда не следует успокаиваться на достигнутом В связи с этим вспоминается следующая поучительная история из жизни физиков на рубеже между XIX и XX вв.

Макс Планк после окончания университета обратился за советом к одному маститому профессору и сказал ему, что решил заняться теоретической физикой

«Молодой человек, – ответил профессор, – зачем вы хотите испортить себе жизнь, ведь теоретическая физика в основном закончена Стоит ли браться за такое бесперспективное дело?!».

В то время на ясном небосклоне физики появились два небольших облачка, из которых спустя несколько лет родились квантовая теория, основы которой были заложены собственно Планком, и теория относительности, созданная никому не известным молодым техническим экспертом III класса, работавшим в Бернском бюро патентов, Альбертом Эйнштейном.

Возможно, такими облачками в дальнейшем развитии компьютеров являются: функциональная электроника и, особенно, нано- и биоэлектроника.

Список сокращений

АЛУ – арифметическо-логическое устройство
БЛЭ – базовый логический элемент
ЗУ – запоминающее устройство
ЛКМ – левая кнопка мыши
ЛФ – логическая функция
ЛЭ – логический элемент
ОЗУ – оперативное запоминающее устройство
ПЗУ – постоянное запоминающее устройство
ПКМ – правая кнопка мыши
ПЛИС – программируемые логические интегральные схемы
ПЛМ – программируемые логические матрицы
ПМЛ – программируемая матричная логика
СИД – светоизлучающий диод
ТИ – таблица истинности
ТТЛ – транзисторно-транзисторная логика
УГО – условное графическое обозначение
ЦУ – цифровые устройства
ЯП – ячейка памяти

• • •

ANSI (American National Standard Institute) – стандарт США
BSC (Bit Storage Cell) – двоичная запоминающая ячейка
DIN (Deutsche Ingenieuring Normen) – немецкий инженерный стандарт
GND (Ground) – заземление
H (High) – высокий логический уровень
L (Low) – низкий логический уровень
LED (Light-Emitting Diode) – светоизлучающий диод
EWB – Electronics Workbench (название программы)
MC – Micro-Cap (название компьютерной программы)
RAM (Random Access Memory) – память с произвольным доступом
ROM (Read-Only Memory) – память только для чтения
PAL (Programmable Array Logic) – программируемая матричная логика
PLA (Programmable Logic Array) – программируемая логическая матрица

Ресурсы Интернет

http://www.softline.ru – официальный сайт дистрибьюторов программы Electronics Workbench

http:// workbench.online.kg/ – файл русифицированной справки Electronics Workbench 5.12

http:// www. rodnik.ru – официальный сайт дистрибьюторов программы Micro-Cap

http:// www. csoft.ru – официальный сайт дистрибьюторов различных программ напрввления САПР

http:// www. chipnews.ru – сайт научно-твхничвского журнала, в котором приводятся новости элвктроники и справочные материалы

http:// www.padio .ru – сайт научно-популярного журнала «Радио»

http:// rl.qrz.ru – сайт научно-популярного журнала «Радиолюбитель»

http:// www. dian.ru – сайт научно-техничвского журнала «Схемотехника»

http:// www. masterkit.ru – сайт разработчиков и дистрибьюторов электронных наборов и модулей для самодвятвльного творчества

Список литературы

1 **Алексенко А Г., Шагурин И.И.** Микросхемотехника – М Радио и связь, 1982 – 416 с
2 **Воробьев Н** Цикл статей по цифровой электронике // CHIP NEWS 1997 – 2000
3 **Джонс М.Х.** Электроника – практический курс – М Постмаркет, 1999 – 528 с
4 **Зельдин Е А.** Триггеры – М Энергоатомиздат 1983 – 96 с
5 **Калабеков Б.А.** Цифровые устройстаа и микропроцессорные системы – М Горячая линия – Телеком 2000 – 336 с
6 **Кардашев Г.А.** Виртуальная электроника Компьютерное моделирование аналоговых устройств – М Горячая линия – Телеком 2002 – 260 с – (МРБ 1251)
7 **Карлащук В.И** Электронная лаборатория на IBM PC Программа Electronics Workbench и ее применение – М Солон-Р 2001 – 726 с
8 **Компьютеры** Спраaочное руководство В 3-х т Т 1 – М Мир, 1986 – 416 с
9 **Микросхемы ТТЛ** Справочник В 2-х т Пер с нем – М ДМК Пресс, 2001
10 **Новиков Ю В** Основы цифровой схемотехники – М Мир, 2001 – 371с
11 **Опадчий Ю.Ф.** и др Аналоговая и цифровая электроника – М Горячая линия – Телеком 1999 – 768 с
12 **Партала О.Н.** Цифровая электроника – СПб Наука и техника 2000 – 208 с
13 **Партала О Н** Цифровые КМОП микросхемы Справочник – СПб Наука и техника 2000 – 393 с
14 **Петцольд Ч.** Код – М Изд -торг дом «Русская редакция» 2000 – 512 с
15 **Применение** интегральных схем В 2-х кн Кн 2 – М Мир 1987 – 411 с
16 **Прянишников В.А.** Электроника – СПб КОРОНА принт 1998 – 400 с
17 **Пухальский Г.И., Новосельцева Т Я** Цифровые устройства – СПб Политехника 1996 – 878 с
18 **Разевиг В.Д.** Система схемотехнического проектирования Micro-Cap 6 – М Горячая линия – Телеком, 2001 – 344 с
19 **Соловьев В.В** Проектирование цифровых систем на основе ПЛИС – М Горячая линия – Телеком 2001 – 636 с
20 **Степаненко И П.** Основы микроэлектроники – М Сов радио 1980 – 424 с
21 **Титце У , Шенк К.** Полупроводниковая схемотехника – М Мир 1982 – 512 с
22 **Токхейм Р.** Осноаы цифровой электроники – М.. Мир, 1988 – 392 с
23 **Тули М.** Справочное пособие по цифровой электронике – М Энергоатомиздат 1990 – 176 с
24 **Угрюмов Е.** Цифровая схемотахника – СПб БХВ – СПб 2000 – 528 с
25 **Уэйкерли Дж.** Проектирование цифровых устройств В 2-х т Пер с англ – М Постмаркет 2002
26 **Фромберг Э.М.** Конструкции на элементах цифровой техники – М Горячая линия – Телеком, 2001 – 264 с – (МРБ 1249)
27 **Хокинг Г.** Цифровая электроника для начинающих – М Мир, 1990 – 232 с
28 **Хоровиц П , Хилл У** Искусство схемотехники В 3-х т – М Мир 1993 – Т 2 3
29 **Шило В.Л.** Популярные микросхемы КМОП – М Горячая линия – Телеком, 2001 – 112 с – (МРБ, 1246)
30 **Электротехника** и электроника в экспериментах и упражнениях Практикум по Electronics Workbench / Под ред Д И Панфилоаа В 2 т – М Додэка 1999 – 2000
31 **Янсен Й** Курс цифровой электроники В 4-х т – М Мир, 1987

Содержание